8-1-1991

# Colorimetric tolerances of digital images

Mike Stokes

# Colorimetric Tolerances of Digital Images

Mike Stokes

B.S. University of Texas at Austin (1989)

A thesis submitted for partial fulfillment
of the requirements for the degree of
Master of Science in Color Science
in the Center for Imaging Science
in the College of Graphic Arts and Photography
of the Rochester Institute of Technology

August 1991

Mike Stokes
Signature of the Author

Mark D. Fairchild
Accepted by
Coordinator, M.S. Degree Program

College of Graphic Arts and Photography
Rochester Institute of Technology
Rochester, New York

CERTIFICATE OF APPROVAL

---

M.S. DEGREE THESIS

---

The M.S. Degree Thesis of Mike Stokes
has been examined and approved
by two members of the color science faculty
as satisfactory for the thesis requirement for the
Master of Science degree.

Dr. Roy Berns, Thesis Advisor

Dr. Mark Fairchild, Thesis Advisor

**Thesis Release Permission Form**

Rochester Institute of Technology
Center for Imaging Science

Title of Thesis Colorimetric Tolerances of Digital Images

I, Mike Stokes, hereby grant permission to the Wallace Memorial Library of R.I.T. to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Date 8/29/91.

# Colorimetric Tolerances of Digital Images

Mike Stokes

Submitted for partial fulfillment
of the requirements for the degree of
Master of Science in Color Science
in the Center for Imaging Science
in the College of Graphics Arts and Photography
at the Rochester Institute of Technology

ABSTRACT

An environment to derive colorimetric tolerances of images was established and an experiment using this new environment was performed. This environment allows for images to be digitally captured, colorimetrically manipulated, displayed, observed, and statistically evaluated. The visual experiment measured perceptibility and acceptability colorimetric tolerances for images using paired comparison techniques. Thirty-two observers judged six typical photographic scenes displayed on a high resolution color monitor. These scenes were manipulated using ten linear and nonlinear functions in the CIELAB dimensions of lightness, chroma, and hue angle. The tolerances were determined using probit analysis. It was found that scene content did not significantly affect the tolerances. The CIELAB, CMC, and MCSL color difference equations were shown to be inadequate for accurately modeling image tolerances. Finally, possible applications of this work are described.

**Acknowledgements**

I am very appreciative of the following sources for support in completion of this thesis :

Hewlett-Packard, Inc. for generously donating equipment used to perform many of the computations in the work,

R. R. Donnelley and Sons, Inc. for the monetary support allowing me the opportunity to study color science,

Franc Grum Memorial Scholarship for monetary support and inspiration in my studies,

Dr. Roy Berns for guidance and always asking the most of me,

Dr. Mark Fairchild for the pep talks, endless editing and encouragement,

each of my observers whose patience and willingness created statistically outstanding results

and Brenda Stokes for not throwing my clothes out the window.

Thank you all very much.

# Table of Contents

# List of Images

# List of Figures

# List of Graphs

# List of Tables

"One conclusion, only, can be stated with assurance, that the successful index of quality of color reproduction will ultimately be established as a result of psychophysical analysis of judgements of picture quality, referred unambiguously to the pictures by measurements of relevant optical quantities." (MacAdam, 1951)

# 1. Introduction

The 1931 CIE standard observer created a unique opportunity for color reproduction to become independent of color devices and media. This established the human visual system as the common denominator between all color reproduction systems. Until then, these systems were forced to be self-contained since no reasonable common denominator between systems existed. Within the next few years Hardy and Wurzburg and MacAdam established the theoretical foundation of colorimetric image reproduction. MacAdam used these tools in 1942 to derive colorimetric tolerances for the standard observer. With these tolerances, it was theoretically possible to determine if an image reproduction was perceptibly different from the original under a given set of observational conditions. It was also theoretically possible to go beyond colorimetric reproduction and derive techniques to compensate for different viewing conditions and illuminants. Together, these results would provide the ability to reproduce acceptable images between different color devices.

Unfortunately, the equipment and technology did not exist until recently to pursue this research. The ability to accurately measure and manipulate each picture element within an image based on the standard observer was required. Such measurements and manipulations were unfeasible given the available technology. Since individual pixels could not

be manipulated, a few relatively large, uniformly colored patches were used to approximate pictorial images. Today such collections of colored patches are called "color charts." These charts are commonly used in the printing, photographic, and television industries for developing both color tolerances and color reproduction techniques. Yet these charts are only an approximation of true images and are known to be very sensitive to the particular colors in the chart. This meant that a reproduction system could be set up to reproduce the color chart well, but not be able to create an acceptable reproduction of some original images. Users of color charts must also assume falsely that observers view uniform patches of color and pictorial images identically. This false assumption leads to additional color reproduction difficulties.

The lack of adequate computational resources made using the standard observer as the least common denominator between systems impractical. Therefore most color reproduction research varied the production controls on the particular systems being studied. In order to be more practical, media dependent measurements of these control variations were made, such as status densities, equivalent neutral densitities, and Institute of Radio Engineers' units. This forced most color reproduction systems to be self-contained.

Successful color image reproduction between different devices or media proved even more formidable. As new cross-media systems were required, completely new resources were created. Numerous negative film stocks were produced in order to deal with the various reproduction needs of the photographic industry, such as printing onto paper, printing onto motion picture film and printing onto graphic arts printing plates. In order to convert motion pictures into a format suitable for the television industry, the

entire telecine industry was created. These systems proved very successful and provided acceptable reproductions. This success was due in no small part to the fact that the original images were almost never seen next to the reproduction.

In the intervening years, most industries involved in color reproduction created an almost endless list of heuristic techniques that were used to obtain acceptable color reproduction. Many of these techniques deal with gamut mapping, chromatic adaptation, and viewing condition parameters. A yardstick was needed to determine which techniques were visually superior. Otherwise, psychophysical experiments must be run for each set of techniques.

The advent of computer workstations and digital color imaging systems in the late 1980's finally created the ability to manipulate each picture element and to use the standard observer in image manipulations. Several researchers used these systems to characterize various digital color input/output devices and created the ability to accurately manipulate each picture element. With these new systems and the characterization tools, it was finally possible to explore the theoretical foundations established over fifty years ago.

This thesis explored these foundations. An environment was established to derive colorimetric tolerances of images and an experiment using this new environment was performed. This environment allows for images to be captured, colorimetrically manipulated, displayed, observed, and statistically evaluated pixel by pixel. The visual experiment measured perceptibility and acceptability colorimetric tolerances for images using paired comparison techniques. Perceptibility and acceptability tolerances were measured in order to provide this visual yardstick.

## 2. Background

A brief historical review of color tolerances in various industries provides significant insight into the current status of digital color reproduction. Color tolerances and color reproduction methods are significantly intertwined. In order to judge whether an image reproduction is acceptable or even perceptibly different, either the reproduction is compared against previously established tolerances or psychophysical experiments are used to directly compare the original and the reproduction.

The paint, textiles and plastics industries use colorimetric methods in determining tolerances for uniform fields of color. Significant progress has been made in this area since this is equivalent to a single pixel calculation. The development and current status of colorimetric tolerances provides possible insight into future directions for the color reproduction of images. The extensive research in chromatic adaptation and environmental effects provide an effective starting point for similar research using pictorial images.

The photographic, printing, and television industries do not have the luxury of dealing with single pixels and thus have been forced to create innovative solutions to the problems of color reproduction. The lack of computational power and input/output devices was the major limitation in this development. The photographic industry is the most technically developed and provides the clearest indication of an underlying pattern in the development color reproduction techniques. This pattern consists of three phases: objective and instrumental development, subjective and psychophysical development, and finally device independent development. The printing and television industries reflect this same pattern to a less developed degree. Examining this pattern help illustrate both the current

status of color reproduction and the problems in overcoming this status. In addition, this background illustrates the depth that current imaging industries are vested into device dependent or simplified colorimetric techniques.

## 2.1 Colorimetric Tolerances in Paint, Textiles and Plastics

Colorimetric tolerances have a long history in color science. Following the CIE creation of a standard observer in 1931, two large bodies of tolerance data were established using uniform fields of color. MacAdam established a large body of threshold data by measuring perceptibility ellipses in 1942.[1] A second body data for large color differences was established by the Optical Society of America Subcommittee on Spacing of Munsell Colors.[2,3] These two bodies of data provided a foundation from which several color difference formulas were developed. Color difference formulas were used to progress beyond simple color matching methods. These formulas provided a basis for color tolerances by quantifying the differences between colors. By the early 1970's, over twenty color difference formulas were in active use. In 1976, the CIELAB and CIELUV color difference formulas were adopted as international standards.[4] While this standardization simplified the choice of color difference formulas, serious deficiencies were already known to exist in these formulas. The main deficiency being that both CIELAB and CIELUV were not uniform enough for critical color difference work and provided only simplistic chromatic adaptation models. Despite these problems, the CIELAB color difference equation has found widespread acceptance in the paint, plastic, and textile industries.

In the decade following the CIE standards, textile researchers in Great Britain established the JPC79[5] and CMC[6] color difference formulas. The JPC79

color difference equation was one of the first color difference equations to be published by the textile industry in an attempt to improve upon the CIELAB formula and to provide a measurement of acceptability in addition to perceptibility measurements. After several problems were documented with this new formula, the Society of Dyers and Colourists' Colour Measurement Committee (CMC) decided to create an improved color difference equation. The CMC color difference formula has found great success in industry and is an enhancement of the CIELAB color difference formula.[7] Yet several criticisms of this formula have been made, most importantly the lack of theoretical basis and coherency in its data base.[8]

Research at the Munsell Color Science Laboratory has been undertaken to overcome these criticisms by establishing a coherent set of color difference data.[9] These data have been employed to derive a new color difference formula named the MCSL color difference equation in this thesis. The MCSL color difference equation also offers a significant enhancement of the CIELAB formulas and maintains a simplified theoretical basis from a coherent data base.

Together, the CIELAB, CMC and MCSL formulas are under current consideration for standardization by the CIE committee on color difference equations. In addition, the CIELAB and CMC equations remain the most active in use by the paint, plastic and textile industries.

Several researchers have developed models to extend color difference tolerances to account for chromatic adaptation and environmental effects such as surround and illumination levels. Chromatic adaptation is a mechanism that responds to changes in the color of the illuminant when viewing samples. Currently, Hunt[10], Nayatani[11], and Fairchild[12], have published extensive models in this area.

In summary, the major strength of colorimetric methods is device independence. Device independent colors are usually defined in terms of the CIE standard observer and not some device dependent colorant concentration or phosphor digital count. The advantage of device independence is each device need only be characterized to the standard observer and not to every other possible device colorant or phosphor in the reproduction system. Being limited to uniform fields of color is the major weakness of colorimetric tolerances. While collections of uniform fields can help simulate a possible color image, the tolerances are very sample dependent and the choices of what and how many colors is arbitrary. Due to these facts, colorimetric tolerances have found wide use in the paint, textile and plastics industries where most colors are viewed alone or in small numbers. In the photographic, printing and television industries, where images consistent of hundreds or thousands of colors viewed simultaneously, colorimetric tolerances are not as predominant. By understanding the current state of colorimetric methods, they can be directly applied to digital image reproduction.

## 2.2 Color Tolerances of Images

### 2.2.1 Tolerances in the Photographic Industry

Techniques for measuring color image tolerances have been well established in photographic science and are historically divided into two time periods. The first period stressed the objective aspects of image reproduction, dealing mainly with issues internal to the media such as spectral sensitivities and inter-image effects. The second period dealt mainly with issues external to the media, such as viewing conditions. Recently, a third period has begun,

in which researchers are trying to incorporate the first two periods into electronic image reproduction.

In the late 1920's, Jones quantified tone reproduction tolerances for black and white images.[13] He established these tolerances by creating a series of graphs with each graph illustrating a single step in the reproduction process. Tolerances were established by varying exposure levels and measuring grey scales of the images visually judged to be "just barely acceptable."[14] As color photography matured, prevalent research based tolerances on densitometry and dye concentration errors. Densitometry is based on a set of standard separation filters with density being the negative log of transmittance. This technique is a general approximation, since most film dye spectral transmission characteristics do not match the standard filter set. Dye concentrations were usually specified in percentages and derived by unbuilding a film emulsion. While an accurate model predicting the amount of colorant in an emulsion is possible, the characterization data are difficult to collect and the computations are lengthy. As a justification for using filter densities, Yule stated[15] that "the colorimetric method has the disadvantage that workers in the field of color photography do not usually think in terms of colorimetry."

Other reasons for not using colorimetric methods were that the errors of color reproduction could not be calculated theoretically, while filter densities and dye concentrations were more suitable "for the evaluation of actual practical tests", and the colorimetric calculations were relatively difficult according to Yule.[15] Following his lead, a great deal of practical research was accomplished in color photography including research on exposure computation, masking, colored couplers and interimage effects.[16-19] These advances have been instrumental in making photography a popular

and viable industry, yet most of them are very media dependent. In silver halide photography during this era, the image capturing media and the image display media were either the same media in the case of reversal film or matched in the case of negative/positive film systems and thus device independence was not an issue. The common thread throughout this research was the emphasis on the objective aspects of image reproduction with instrumental measurements and tolerances.

By the late 1950's, psychophysical methods were being used to study some of the subjective aspects of image reproduction.[20] These subjective aspects could not be accounted for by objective instrumental measurements. Psychophysical methods involved matching human responses to instrumental measurements in order to derive the preferred reproduction characteristics as opposed to some objective optimal characteristics. Bartleson and Woodbury published several studies on psychophysical methods, establishing basic psychophysical experimental techniques relevant to this industry.[21-24] Following the physical measurements made by Jones and Condit in 1941,[25] many experiments were performed to determine the psychophysical effects of tone reproduction and illumination in color reproduction.[26-34] These works established modern-day tone reproduction theory for color photography. Modern-day tone reproduction incorporates the tone reproduction theory established by Jones with the results of the psychophysical experiments, producing a preferred reproduction image. Other psychophysical experiments investigated tolerances for color casts,[35] surrounds,[36] and comparison of prints and transparencies.[37] All of these results were incorporated into improving various film and print production in the photographic and printing industry. Finally, computational models were created to derive color tolerances for images and thus allow researchers

to simulate the entire reproduction process without the expense of creating new products and testing them.[38,39] With the exception of work by Smits and Corluy,[38] these experiments typically used device or media dependent units such as filter density units. Again, during this time, most film systems were self-contained so this was not a significant problem.

In order to reproduce important device independent colors, several studies were performed on tolerances of memory colors for skin tones, blue sky and green grass.[40-42] These studies were often hybrid experiments, deriving tolerances of uniform fields within a static background image. The tolerances were used as test points for complex imaging systems. If these test points did not fall within acceptable tolerances, the entire system was unacceptable. While this work provided device independent tolerances for a few colors, it did not provide overall tolerances for color images.

### 2.2.2 Tolerances in the Printing Industry

The history of color tolerances in the printing industry follows a similar trend. Building on the basic work of Jones, Neugebauer described the fundamental equations for half-tone printing.[43] Yet this model is not robust enough to predict the system behavior accurately. Most measurements and models were made relative to media dependent units such as filter density or ink concentration.[44-47] Additional problems in the printing industry were addressed by these models, including ink trapping, and half-tone screen angles. Recently, more work has been done to implement colorimetric measurements in tolerances using uniform fields for Standard Web Offset Press (SWOP) inks and the visual inspection process.[48] Currently, tolerances for printing systems and even color electronics pre-press systems (CEPS) are still measured in media dependent units.[49] Until an accurate model for color

half-tone printing can be established, it will remain difficult to deal with issues external to the media, such as viewing conditions. Work is currently being done to both derive an accurate model and to compensate for the external issues.[50,51]

### 2.2.3 Tolerances in the Television Industry

The color television industry generally uses the CIELUV color difference equation with a set of uniform fields to establish color tolerances. Color television was established incorporating the principles of colorimetry.[52–56] The original NTSC characteristics of the sensitivities of television cameras based on the representative television monitors were linear transformations of the CIE standard observer. If all of the equipment and environmental conditions were carefully calibrated, this allowed for accurate colorimetric color reproduction within the television monitor gamut. While this theoretical optimization is possible, it has not been practical due to changes in both the camera sensitivities and the monitor phosphors. Today many of the recommended tolerance techniques use sets of uniform fields.[57] Television engineers approximate full-color images by using a carefully chosen set of uniform fields. The results of these techniques can then be expressed either graphically or numerically.[58-60] These results are very dependent on the chosen samples and there is no consensus on the ideal set of samples.[61,62] Common practice is to include a series of neutrals, critical colors and saturated colors. The neutrals help to maintain a neutral scale in the reproduction. By oversampling colors such as skin tones, green grass and blue sky, certain known critical colors are maintained. Finally, saturated colors are sampled to establish the overall gamut of the reproduction system. There is no accepted standard set of colors since this technique is only an

approximation of actual images and no accepted standard for colorimetric tolerances in the television industry exists.

### 2.2.4 Summary of Color Tolerances of Images

The ability to independently manipulate individual pixels in an image is a major strength of many of these techniques, while utilizing device or media dependent measurements is a major weakness. Due to these facts, image tolerance techniques have found wide use in the photographic and printing industries where most systems are self-contained or allow wide tolerances. Examples of such systems include the motion picture industry where the film families are matched together, the catalog publishing industry where the scanners are matched to particular printing presses and ink sets, and the slide industry where the capturing media and display media exist in the same emulsion. While these systems perform well under the conditions for which they were created, when images are digitally transferred from one system to another, the results are often unacceptable and sometimes disastrous.

## 2.3 Colorimetric Tolerances in Images

### 2.3.1 Theoretical Background

The theoretical basis for overcoming the weaknesses of both colorimetric tolerances and color image tolerances was established several decades ago. Unfortunately, it hasn't been practical to fully implement until recently. Hardy and Wurzburg published the basic colorimetry needed for continuous tone scanning and printing in 1937. MacAdam stated in 1938 that colorimetry could not be ignored with impunity and proceeded to describe in detail the necessary colorimetric theory for photography.[63,64] In the mid-

1950's MacAdam[65] and Neugebauer[66] separately detailed research projects to measure both the objective and subjective aspects of image reproduction and encouraged researchers to do so, but little was published in direct response. During the intervening years, many have repeated their warnings in various ways.[67-70] While color reproduction devices were self-contained systems, device independence was not a strong issue and was therefore ignored by most manufacturers.

### 2.3.2 Example of Device Independent Reproduction System

With the advent of electronic imaging, where images can be scanned, stored, manipulated, displayed and printed on several different systems, device independence is a strong issue as illustrated in figure 1.3-1. In this example, an advertising firm in New York City prepares artwork, electronically digitizes this work, performs some electronic enhancements and finally electronically transfers the digitized image to the client in Los Angeles for approval. The client in Los Angeles views the digital image on a monitor or possibly prints out a copy of the image for approval. Upon approval, a copy of the image is sent to Chicago to be published. The publisher creates printing plates from the digital image and prints four-color reproductions. With currently implemented technology, none of the images will match each other. This is mainly because the images are stored in device dependent units Using device dependent units of color, each device must scan or display the same color, given identical device units. To accomplish this, the sensitivities of the scanner must match the phosphors of the monitor in New York City which must match the phosphors of the monitor in Los Angeles which must match the colorants in the printer in Los Angeles which finally must match the colorants in the printing press in Chicago. This

is never the case and additionally there has been no compensation for different viewing conditions such as different lighting levels, fluorescent lighting versus tungsten lighting and the surround of the image. In order to begin to resolve these conflicts, methods for colorimetrically characterizing devices and measuring colorimetric tolerances for entire images need to be established. By establishing colorimetric tolerances, the necessary accuracy for device characterization will be established. Changing from device dependent color manipulations over to device independent manipulations is a phenomenally costly proposal. Yet, there is little alternative if the situation described above is to be achieved.

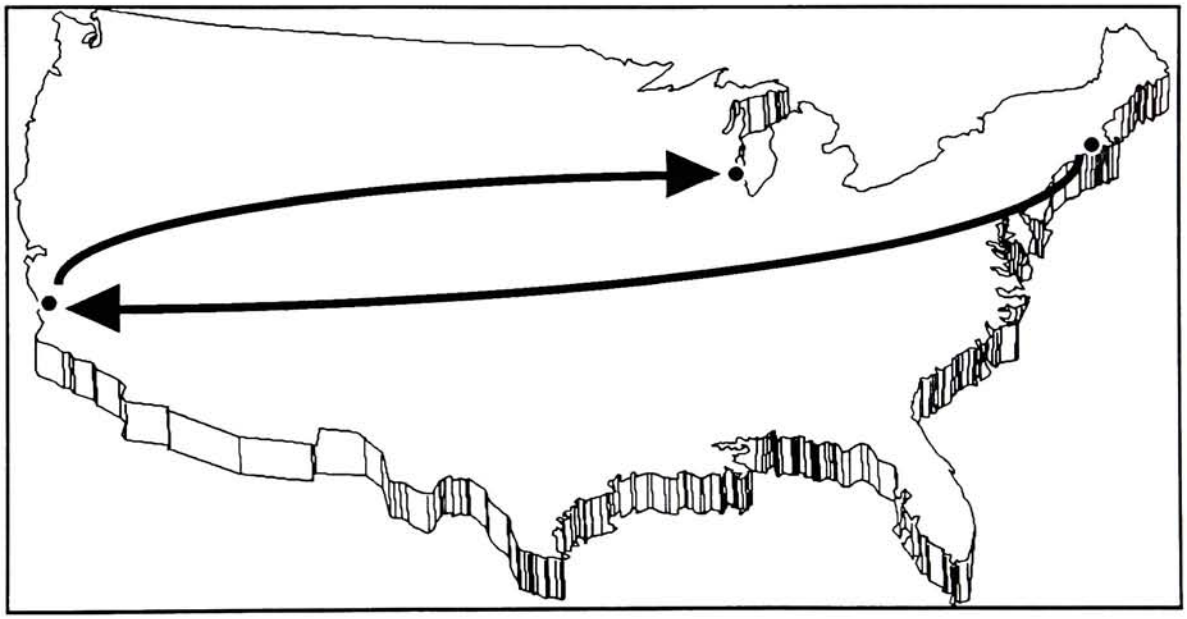


Figure 1.3-1 : An Example of Electronic Imaging

## 2.4 Measuring Colorimetric Tolerances in Images

With the advent of recent digital hardware, it is finally possible to manipulate individual pixels colorimetrically. The last decade has given birth to both 24-bit frame buffers and color graphics workstations. Until the

advent of high speed workstations, with 24- to 36-bit color frame buffers, deriving colorimetric tolerances of images was very difficult since individual color pixels had to be independently manipulated.' This difficulty was due to the size of each pixel being extremely small and in many cases, such as with printing plates, individual pixels could not be independently manipulated. Additionally, 24- to 36-bit frame buffers are necessary to obtain a sufficient number of colors for high quality reproductions. Computer graphics workstations provided the computational support and interfaces to efficiently utilize the frame buffers. Finally, new input and output devices are being marketed every year to interface with the 24-bit frame buffers. Using these new tools, an experimental environment was established that allows for digital capture, colorimetric manipulation, display, observation, and evaluation of images.

One aim of the current research is to combine the best aspects of colorimetric tolerance methods and imaging methods to determine device-independent color tolerances for complex stimuli. This thesis describes a computational environment and psychophysical experiment for the measurement of colorimetrically specified perceptibility and acceptability tolerances in pictorial images.

# 3. Environment

## 3.1 Hardware

The experimental environment allows for images to be digitally captured, colorimetrically manipulated, displayed, observed, and statistically evaluated. The equipment for this setup is shown in figure 3.1-1.

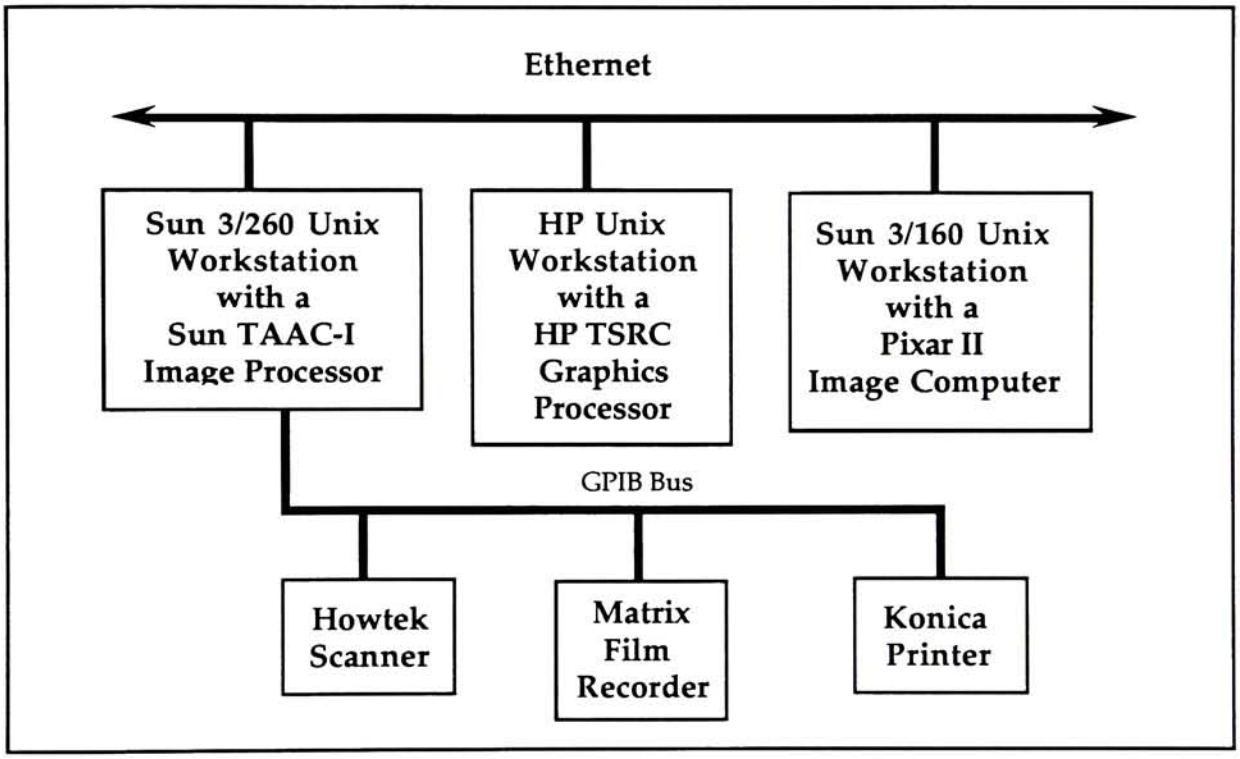



Figure 3.1-1 : Hardware Environment

An ideal color image software package would be able to colorimetrically manipulate a 1280 x 1024 pixel image in real time (less than 1/30 second) with accuracy better than five decimal places in various color spaces. The five decimal places are derived from the four decimal places in very-high-accuracy reflectance factor measurements and the necessary computations to convert these values into colorimetric values while avoiding roundoff errors. With

the hardware currently available to the Munsell Laboratory this speed is completely unfeasible. Table 3.1-1 illustrates this fact by showing the computational speed for various computer workstations. In this illustration, a one thousand by one thousand pixel image was multiplied by a 3x3 matrix using both integer and floating point manipulations. In order to achieve real time performance, approximately 500 MFLOPS would be required. This includes the translation from device space into CIELAB L*C*h$^{o}$, the pixel by pixel color manipulation, the translation back into device space for display and transfering the image between the system memory and the frame buffer. Since our hardware is significantly slower than the real time requirements the experimental images were precalculated for the observations.

Table 3.1-1 : Workstation Performance

| Workstation | MIPS | integer time (secs) | KFLOPS | float time (secs) |
|---|---|---|---|---|
| Sun 3/160 | 2.5 | 56.8 | 70.7 | 222.6 |
| Sun 3/260 | 4.1 | 34.6 | 83.4 | 188.6 |
| HP 9000/375 | 7.8 | 18.2 | 306.0 | 51.4 |
| Sun TAAC I | 5.9 | 24.2 | 650.0 | 24.4 |
| PIXAR II | 31.5 | 4.5 | 35.3 | 445.0 |

The images were digitized using a Howtek/Sharp JX450 flatbed scanner with a custom software interface. A series of hardware options are available including gamma control, edge enhancement, resolution, scanning rate, reflection/transmission modes, positive/negative modes, and custom look-up tables. The interface was written to allow ease of use without losing any of the hardware flexibility. Five optical resolutions are available (300, 200, 150, 75, and 50 dpi) with electronic averaging producing resolutions in integer units between 30 and 300 dpi. Weaknesses of this scanner include a 3:1 signal

to noise ratio and severe spatial non-uniformity problems near either end of the detector array.

Color CRT displays are currently supported in this environment. Calibration includes a white point setup, a linearity calibration, and a primary transformation matrix. The white point setup is achieved by choosing the appropriate white point for the experiment, such as D65 and adjusting the internal monitor calibrations until this balance is achieved throughout the monitor's luminance range. This maximizes achievable luminance range around this white point. The linearity calibration transforms non-linear digital counts into linear RGB signals and is achieved by making a series of measurements for each channel and performing non-linear regression. The primary transformation matrix allows the linear RGB signals to be converted into XYZ tristimulus values and is independent of the setup.[71]

## 3.2 Software

The software development emphasized accuracy, portability, flexibility, and colorimetric device independence. Accuracy is achieved by using 32-bit floating point storage and computations for each color channel, except where hardware is limited, such as at the scanner or display. Portability is achieved by coding in the ANSI C programming language[72] within a UNIX operating system.[73] Flexibility is achieved by using modular code (most routines are less than 60 lines long) and a minimal amount of global variables. Colorimetric device independence is achieved by using CIE tristimulus values[4] as the standard color transformation space. Good documentation is achieved by the liberal use of internal comments, and manual pages for each routine in a style consistent with the UNIX operating system environment.

Numerous image storage formats were reviewed, including IMG,[74] GIF,[75] TIFF,[76] TGA,[77] PIXAR,[78] X11,[79] PPM,[80] and RLE.[81] Almost all of these formats are integer based and thus do not have the accuracy required. Additionally, only the IMG has a white point or any other colorimetric specification. While some provided flexibility, it was decided that this was not adequate and a new storage format was created to provide the precision and flexibility with respect to the colorimetric manipulations. This new storage format, which includes colorimetric information such as color space and white point, has been named "CRS" (color reproduction software) and is described in detail in Appendix A. The PPM format was used to store the raw digitized images and the PIXAR format was used to store the images ready for display.

To create the necessary color space transformations and colorimetric manipulations, a software color reproduction library was created. The main library is divided into six functional sublibraries including color space transformations, colorimetric manipulation functions, file input/output routines, image manipulation functions, color reproduction analysis routines and miscellaneous support functions. Table 3.2-1 lists the function descriptions currently included in the software environment.

Table 3.2-1 : Software Library Routines

| Library Routine | Description |
|---|---|
| calc_dCMC | calculate CMC color differences for a pair of pixels |
| calc_dE | calculate CIELAB color differences for a pair of pixels |
| calc_dMCSL | calculate MCSL color differences for a pair of pixels |
| calc_dMCSL2 | calculate MCSL2 (including non-linear chroma modifier) color differences for a pair of pixels |

| | |
|---|---|
| calc_stat | calculate statistics for a pixel |
| copy_crsh | copy crs image header |
| crs_clear | clear a crs image and header |
| crs_crop | crop a crs image |
| crs_cst | convert a crs image from one color space to another color space |
| crs_error | report error and exit from a crs library routine |
| crs_flipx | flip crs image horizontally |
| crs_move | move crs image within active window |
| crs_scale | scale crs image within active window |
| crs_warn | report warning and return from a crs library routine |
| crs2hp | convert crs image into hp starbase image |
| crs2mds | convert crs image into mds (working format) image |
| crs2pix | convert crs image into raw pixar image |
| crs2ppm | convert crs image into ppm (pbmplus) image |
| crs2xwd | convert crs image into X11 dump image |
| exec_3x1 | execute a 3x1 matrix on a crs pixel |
| exec_3x3 | execute a 3x3 matrix on a crs pixel |
| exec_clip | execute a clipping funciton on a crs pixel |
| exec_crtm | execute a CRT gamma-offset-gain model on a crs pixel |
| exec_dCMC | execute a CMC color difference function on a crs pixel |
| exec_dE | execute a CIELAB color difference function on a crs pixel |
| exec_dMCSL | execute an MCSL color difference function on a crs pixel |
| exec_dMCSL2 | execute an MCSL2 color difference function on a crs pixel |
| exec_hist | execute a histogram function on a crs pixel |
| exec_imfact | execute an inverse multiplicative factor function on a crs pixel |
| exec_mfact | execute a multiplicative factor function on a crs pixel |
| exec_pimf | execute a power/inverse multiplicative factor funciton on a crs pixel |
| exec_power | execute a power function on a crs pixel |
| exec_quant | execute a quantizer function on a crs pixel |
| exec_scrv | execute an sigmoidal function on a crs pixel |
| exec_set | execute a constant function on a crs pixel |
| exec_shps | execute a transfer function on a crs image |

| | |
|---|---|
| exec_stat | execute a simpel statistical functions on a crs pixel |
| exec_tools | execute an analytical tool on a crs image |
| exec_voff | execute an additive offset function on a crs pixel |
| exec_wrap | execute a wraparound function on a crs pixel |
| hp2crs | convert an hp starbase image into a crs image |
| hp2mds | convert an hp starbase image into an mds (working format) image |
| in_crshdr | input a crs image header |
| in_mdshdr | input an mds image header |
| in_ppmhdr | input a ppm image header |
| in_xwdhdr | input an X11 image header |
| init_crs | initialize a crs image |
| init_dCMC | initialize a CMC color difference function |
| init_dE | initialize a CIELAB color difference function |
| init_dMCSL | initialize an MCSL color difference function |
| init_dMCSL2 | initialize an MCSL2 color difference function |
| init_hist | initialize a histogram function |
| init_mds | initialize an mds image |
| init_pix | initialize a pixar image |
| init_ppm | initialize a ppm image |
| init_shps | initialize a transfer function |
| init_stat | initialize a statistical function |
| Lab_LabLCh | convert a CIELAB pixel into a CIELAB L*C*h° pixel |
| Lab_XYZ | convert a CIELAB pixel into a tristimulus pixel |
| LabLCh_Lab | convert a CIELAB L*C*h° pixel into a CIELAB pixel |
| Luv_LuvLCh | convert a CIELUV pixel into a CIELUV L*C*h° pixel |
| Luv_XYZ | convert a CIELUV pixel into a tristimulus pixel |
| LuvLCh_Luv | convert a CIELUV L*C*h° pixel into a CIELUV pixel |
| load_3x1 | load a 3x1 matrix function |
| load_3x3 | load a 3x3 matrix function |
| load_cal | load CRT gamma-offset-gain calibration data |
| load_clip | load a clipping function |
| load_crs | load a crs image from disk |
| load_crsh | load a crs image header |
| load_crsi | load a crs image data |
| load_crtm | load a CRT gamma-offset-gain function |
| load_imfact | load an inverse multiplicative factor function |
| load_lmts | load crs color space dimensional limits |
| load_mds | load an mds image from disk |

| | |
|---|---|
| load_mfact | load a multiplicative factor function |
| load_null | load a null function |
| load_pimf | load a power/inverse multiplicative factor function |
| load_power | load a power function |
| load_ppm | load a ppm image from disk |
| load_quant | load a quantizer function |
| load_rgb | load a raw rgb image from disk |
| load_scrv | load a sigmoidal function |
| load_set | load a constant function |
| load_voff | load an additive offset function |
| load_win | load an active window |
| load_wrap | load a wrap-around function |
| load_xwd | load an X11 image from disk |
| mds2crs | convert an mds image into a crs image |
| mds2hp | convert an mds image into an hp starbase image |
| mds2pix | convert an mds image into a pixar image |
| mds2ppm | convert an mds image into a ppm image |
| mds2xwd | convert an mds image into an X11 image |
| ppm2crs | convert a ppm image into a crs image |
| ppm2hp | convert a ppm image into an hp starbase image |
| ppm2xwd | convert a ppm image into an X11 image |
| readrow | read a single row of a crs image |
| reset_win | reset active window |
| save_crs | save a crs image to disk |
| save_crsh | save a crs image header |
| save_crsi | save crs image data |
| save_mds | save an mds image to disk |
| save_pix | save a pixar image to disk |
| save_ppm | save a ppm image to disk |
| save_xwd | save an X11 image to disk |
| scn_XYZ | convert a raw rgb scanner pixel into an XYZ tristimulus pixel |
| writerow | write a single row of a crs image |
| XYZ_crt | convert an XYZ tristimulus pixel into a CRT rgb pixel |
| XYZ_Lab | convert an XYZ tristimulus pixel into a CIELAB pixel |
| XYZ_Ljg | convert an XYZ tristimulus pixel into an OSA Ljg pixel |
| XYZ_Luv | convert an XYZ tristimulus pixel into a CIELUV pixel |

| | |
|---|---|
| XYZ_YIQ | convert an XYZ tristimulus pixel into an NTSC YIQ pixel |
| XYZ_Yxy | convert an XYZ tristimulus pixel into a Yxy chromaticity pixel |
| xwd2mds | convert an X11 image into an mds image |
| xwd2ppm | convert an X11 image into a ppm image |
| YIQ_XYZ | convert an NTSC YIQ pixel into an XYZ tristimulus pixel |
| Yxy_Ljg | convert a Yxy chromaticity pixel into an OSA Ljg pixel |
| Yxy_XYZ | convert a Yxy chromaticity pixel into an XYZ tristimulus pixel |

Currently supported color spaces include : XYZ tristimulus space, Yxy chromaticity space, CIELAB L*a*b*, CIELAB L*C*h$^{o}$, CIELUV L*u*v*, CIELUV L*C*h$^{o}$, YIQ space,[82] HLS space,[83] and device dependent spaces in RGB coordinates. The XYZ tristimulus space is used as the common transfer space. Image data in any other space can be transformed to and from XYZ. This allows any one space to be transformed to any other space as shown in figure 3.2-1. This flexibility also reduced the code complexity, a factor that outweighed the decrease in execution speed.

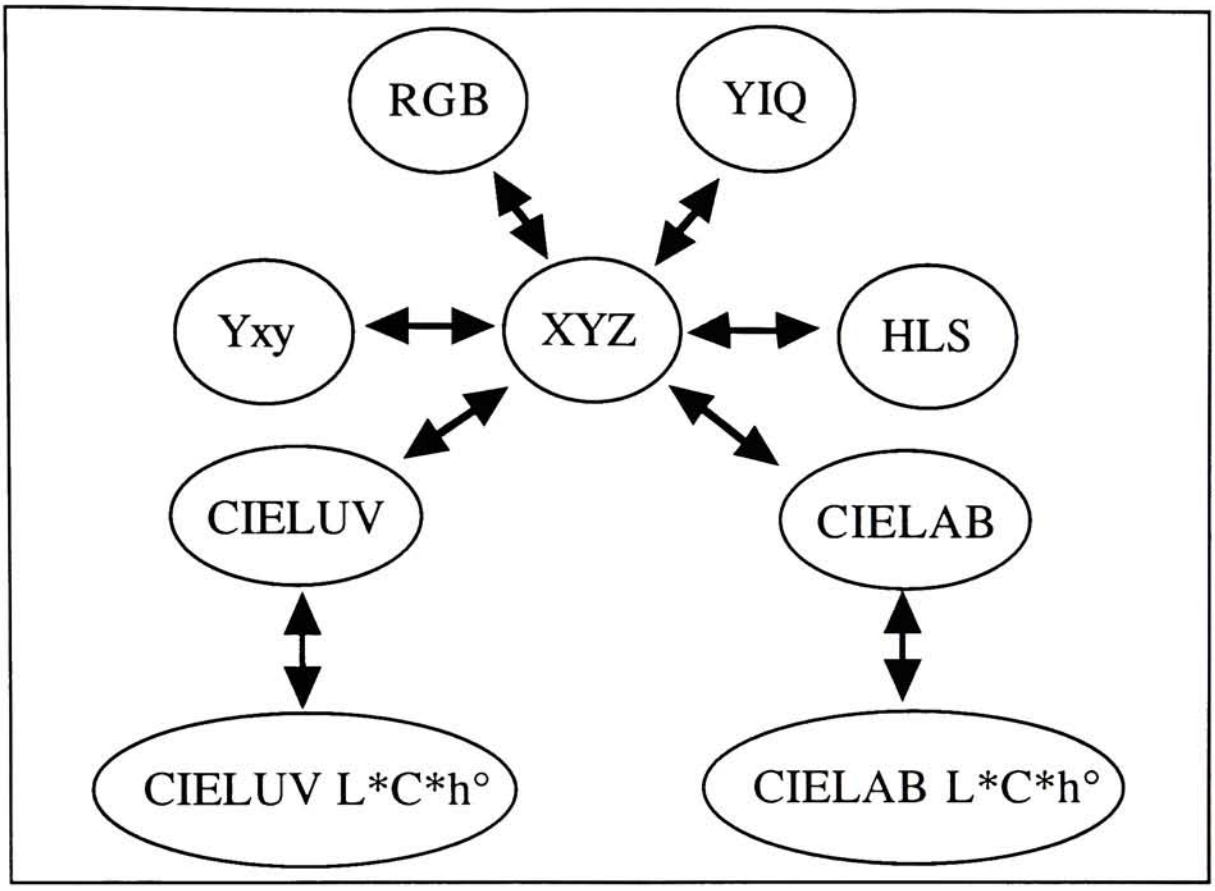



Figure 3.2-1 : Color Space Transformations

Transfer functions are simple mathematical functions relating an output image to the original input image.[84] A number of transfer functions are currently implemented including a set constant, an additive offset, a multiplicative factor, a multiplicative factor with an additive offset, a power function combined with a multiplicative factor with an additive offset, a power function, a sigmoidal function, a quantizer function, matrix functions, a gamma-offset-gain modeling function and boundary functions.

A variety of other support routines include device dependent storage routines, color reproduction analysis routines, various input/output routines, and miscellaneous support routines. Currently implemented

storage formats include X11, Pixar, raw data and a few custom formats.[85] Analysis routines include histograms, statistics, and the CMC,[6] MCSL,[9] and CIELAB[4] color difference equations.

# 4. Experimental

The environment described above was used to measure perceptibility and acceptability tolerances of images. The images were manipulated using four general transfer functions in the CIELAB dimensions of lightness, chroma, and hue angle.

## 4.1 Choosing Optimal Dimensions

The criteria for choosing optimal dimensions included visual uniformity, intuitiveness, industry standardization, and applicability to current color difference formulas. The CIELAB dimensions of lightness, chroma, and hue angle, were chosen as the most appropriate dimensions to meet these criteria. Munsell and others before him illustrated that the dimensions of lightness, chroma and hue are very intuitive.[86] Common spaces in current use, such as YIQ or HLS, provide no device independence and are significantly nonuniform. The Yxy chromaticity, while device independent, is not perceptually uniform. CIELUV,[4] while meeting most of these criteria, was not as applicable to current color difference equations.

## 4.2 Transfer Functions

Four transfer functions were used to manipulate the images in the chosen dimensions. The functions are fundamental mathematical contructs and simulate common industry process transformations for contrast,[87] gain, gamma controls,[88] and color casts or shifts.[35] The functions were applied to the dimensions of lightness, chroma, and hue. The mathematical form of these four transfer functions are : 1) an additive offset, 2) a multiplicative offset, 3) a power function, and 4) a sigmoidal function. The sigmoidal

function was implemented using a combination of two power functions. Mathematical descriptions for these transfer functions are shown below in table 4.2-1 and graphically illustrated in figure 4.2-1.

Table 4.2-1 : Transfer Functions Formulas

| Function | Formula |
|---|---|
| Multiplicative Factor | $Out = In * Constant$ |
| Power | $Out = In^{Constant}$ |
| Sigmoidal | $Out = \begin{cases} \frac{(In*2)^{Constant}}{2} & In < 0.5 \\ \frac{\left[\left[(In*2)-1.0\right]^{1/Constant}\right]+1.0}{2} & In \geq 0.5 \end{cases}$ |
| Additive Offset | $Out = In + Constant$ |

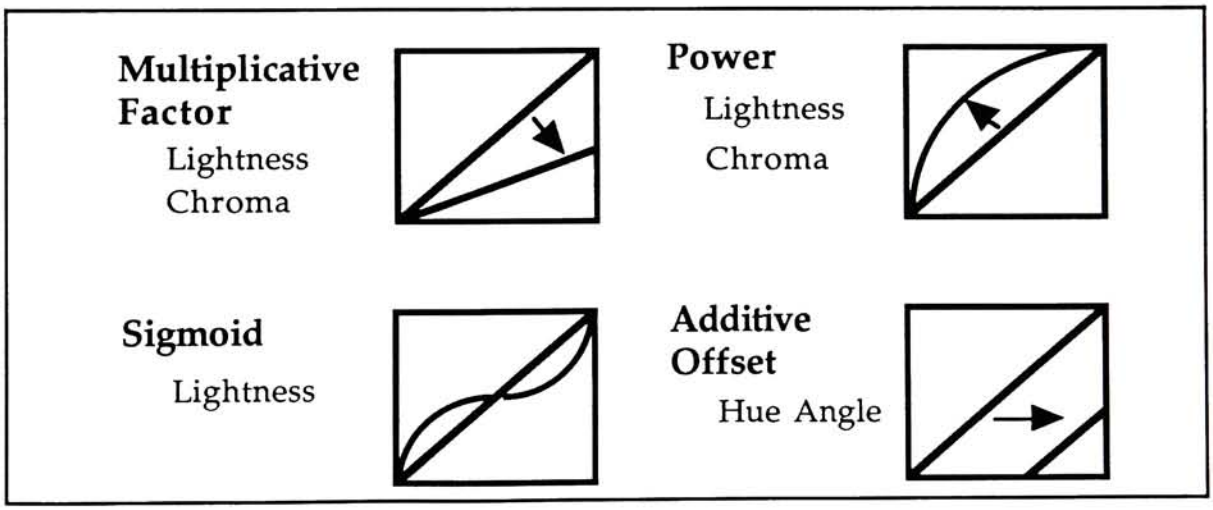



Figure 4.2-1 : Transfer Functions

Given the three dimensions and four functions, a total of twelve combinations exist. Common sense dictated the elimination of several of these combinations. For example it made no sense to have a multiplicative

function applied to hue or an additive offset applied to chroma. The final combinations used in this work were : 1) a multiplicative factor in lightness, 2) a power function in lightness, 3) a sigmoidal function in lightness, 4) a multiplicative factor in chroma, 5) a power function in chroma, and 6) an additive offset in hue. These are summarized in table 4.2-2. Since symmetry could not be assumed for the additive offset, power and sigmoidal functions, each of these functions were divided into either high and low or positive and negative parameter levels. These divisions brought the total number of unique transfer function/color space dimension combinations to ten.

Table 4.2-2 : Implemented Transfer Functions and Dimensions

| Transfer Function | Name | CIELAB Dimension | Parameter Values |
|---|---|---|---|
| Multiplicative Factor | LMF | Lightness | <= 1.0 |
| Power | LPH | Lightness | >= 1.0 |
| Power | LPL | Lightness | <= 1.0 |
| Sigmoidal | LSH | Lightness | >= 1.0 |
| Sigmoidal | LSL | Lightness | <= 1.0 |
| Multiplicative Factor | CMF | Chroma | <= 1.0 |
| Power | CPH | Chroma | >= 1.0 |
| Power | CPL | Chroma | <= 1.0 |
| Additive Offset | HOH | Hue Angle | >= 0.0 |
| Additive Offset | HOL | Hue Angle | <= 0.0 |

General transfer functions do not assume independence.[56] Ideally the dimensions of lightness, chroma and hue are independent of each other. This is not the case in the CIELAB color space, as documented by the Helmholtz-Kohlrausch effect[89] and nonlinear hue lines.[90] It was assumed that the interdependence between lightness, chroma and hue was minimal.

Some of the transfer functions required anchor points to be set. Anchor points are points in a transfer function that are stable with respect to

changes in the function's parameters. The anchors must be chosen in context of the dimension being shaped. For example, a power transfer function for L* has two canonical anchors at the values of 0 and 100. For chroma, the maximum (or even mid-point) anchor is not intuitive. These examples are illustrated in figure 3.2-2.

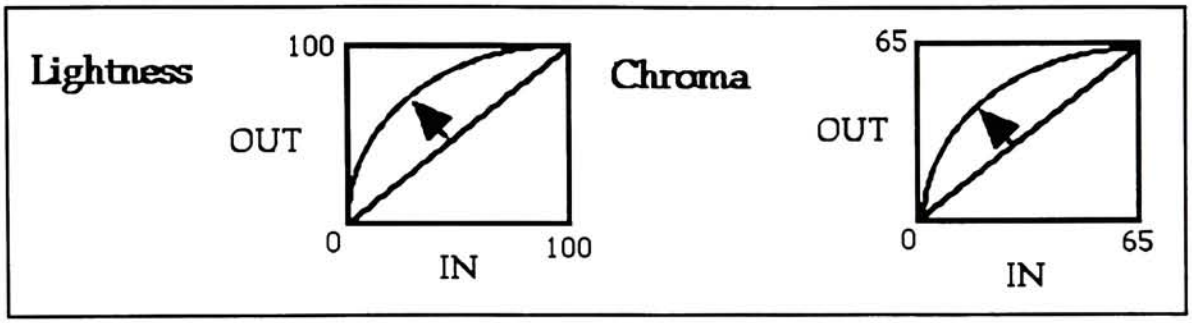



Figure 4.2-2 : Lightness and Chroma Anchors

In this experiment, the maximum chroma anchor was set to 65 (or about twice the average of blue sky, green grass and flesh tones).[41] Other possibilities include making the anchor equal to : 1) the maximum of the gamut, 2) the maximum of the gamut at that particular hue and lightness, or 3) the minimum of the maximum of the gamut at that particular hue and lightness, and the maximum of the gamut for the image at that particular hue and lightness. By choosing twice the sky, skin, and grass mean, device independence has been maintained; unfortunately this choice also induced a cylindrical treatment of the chroma boundaries. Table 4.2-3 below describes each transfer function and its anchors.

Table 4.2-3 : Transfer Functions and Anchors

| Transfer Function | CIELAB Dimension | Anchor Values |
|---|---|---|
| Multiplicative Factor | Lightness | 0.0 |
| Power | Lightness | 0.0 & 100.0 |
| Power | Lightness | 0.0 & 100.0 |
| Sigmoidal | Lightness | 0.0 & 50.0 & 100.0 |
| Sigmoidal | Lightness | 0.0 & 50.0 & 100.0 |
| Multiplicative Factor | Chroma | 0.0 |
| Power | Chroma | 0.0 & 65.0 |
| Power | Chroma | 0.0 & 65.0 |
| Additive Offset | Hue Angle | n/a |
| Additive Offset | Hue Angle | n/a |

## 4.3 Image Selection

Three dominant concerns in analyzing pictorial images have been scene content dependence,[25] perceived object distance[91] and overall chroma levels.[20] Six different images were used in order to examine these concerns. The six images were divided into three scene content types and two levels each for perceived object distance and chroma content. The three scene types were man-made objects, people and nature scenes.

With these concerns in mind, several professional photographers were solicited to loan original images to be digitally captured. The final images were selected from several hundred originals based on industry use, scene content, overall chroma level, object distance, professional composition, overall neutral color balance, lighting and exposure range. The originals were transferred from either 4 x 5 inch or 35mm transparencies into 35mm internegatives and finally back to 8 x 10 inch color prints. These extra steps were taken to assure long term availability of high quality images.

Representations of these images are shown below along with the

background images. Due the the reproduction process used, these images serve only as crude representations of the actual images used in the experiment. Edge enhancement was used on image 4 during the scanning process to improve the apparent sharpness.

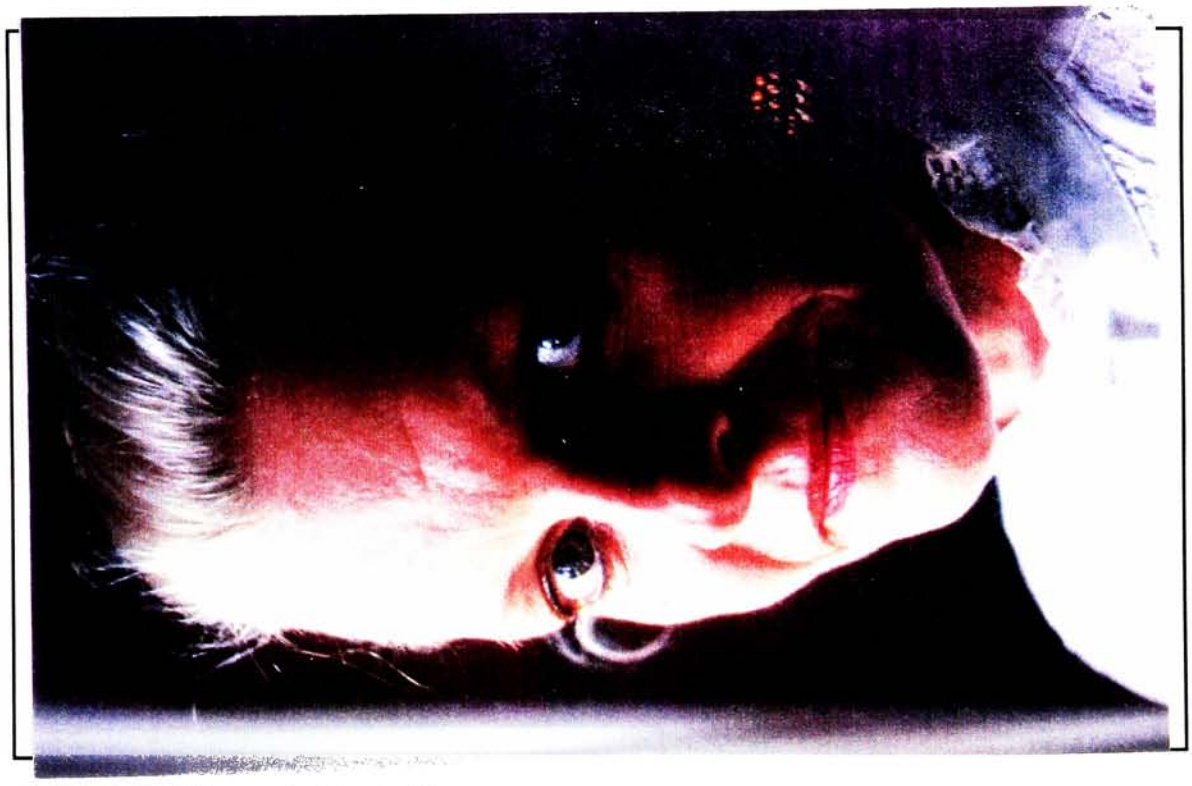


Image 1 : Woman's Portrait

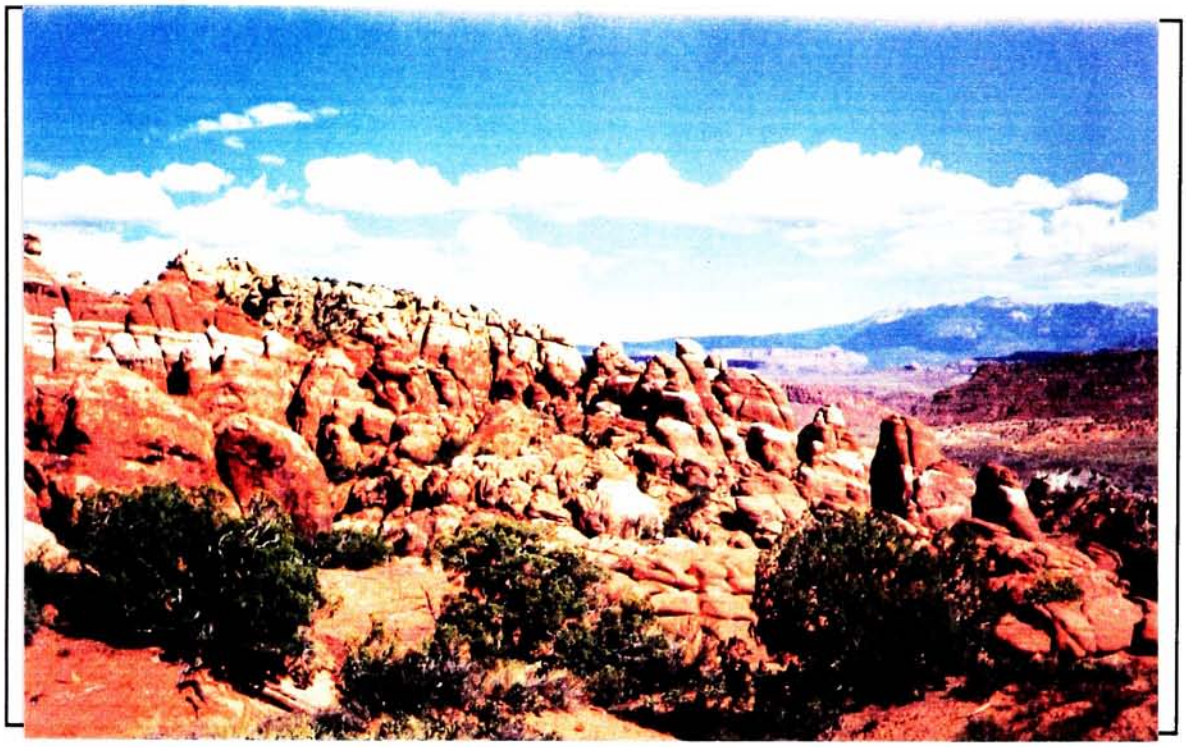


Image 2 : Desert Landscape

Image 3 : Clothing Materials - Kodak Q60

Image 4 : Three Women - Kodak Q60

Image 5 : Perfume Bottles

Image 6 : Tree Leaves

The background images were created by photographing several pairs of hands holding a five by seven white card on a Munsell N5, neutral cardboard background. The final two images, one with the white card placed horizontally and one with it placed vertically, were selected based on the natural appearance of the model's hands. Eight by ten inch prints were made from these 35mm negatives.

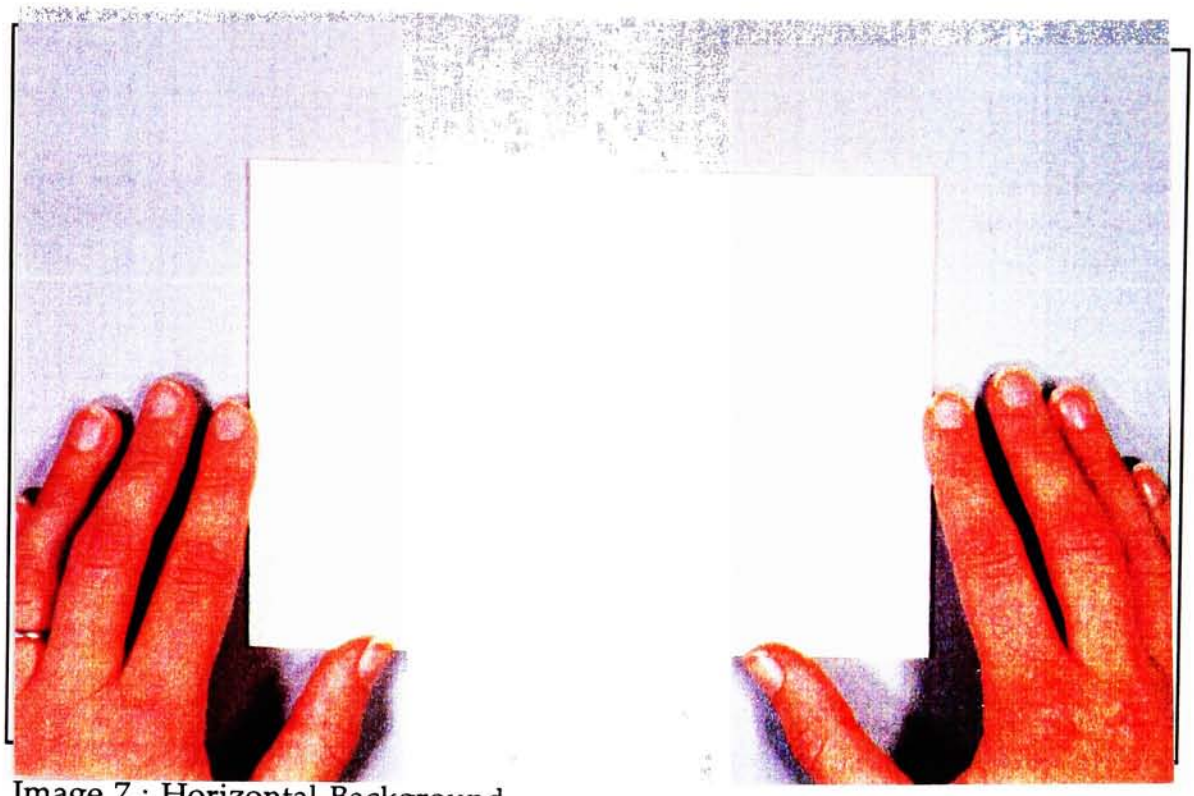


Image 7 : Horizontal Background

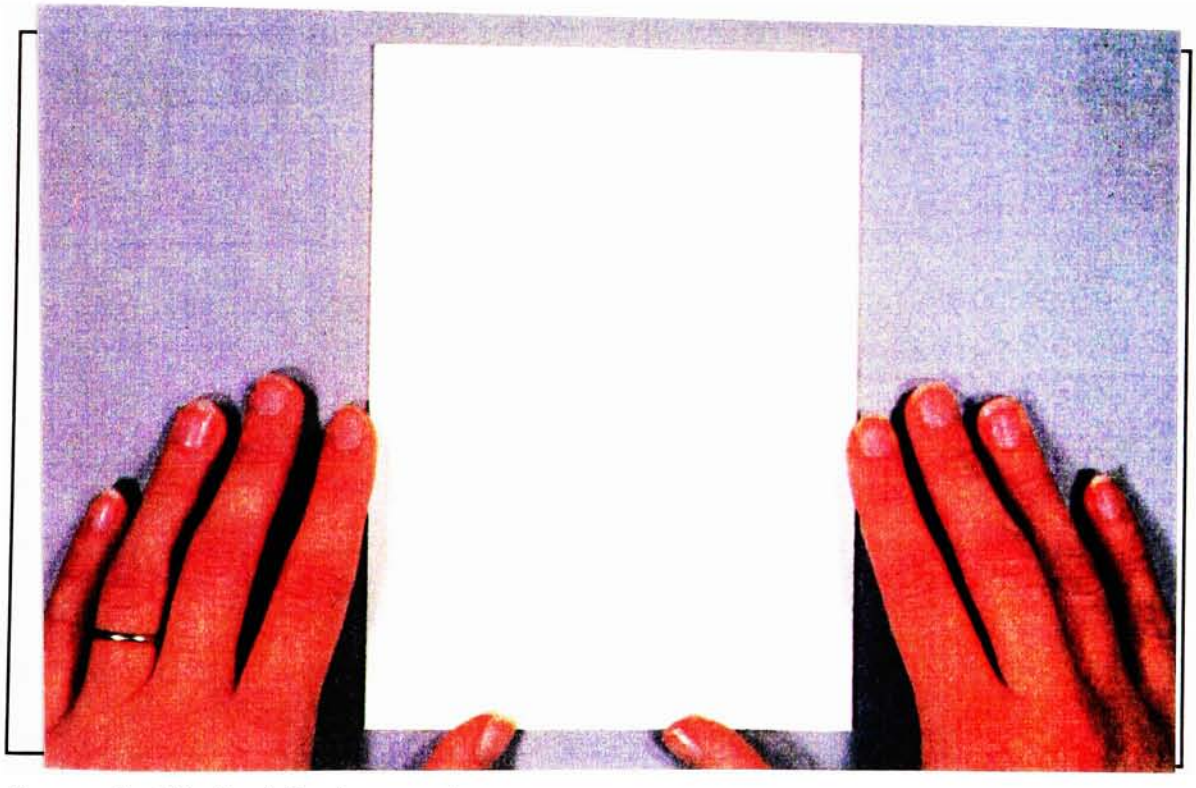


Image 8 : Vertical Background

## 4.4 Image Capture

The eight by ten inch prints were digitized using a Howtek/Sharp JX450 flatbed scanner. Each image was cropped to obtain the best subjective composition. Since the final display image was to be four inches by six inches on the Pixar display, each image was scanned into either 380 by 570 pixels or 570 by 380 pixels depending on the aspect format. The default step scanning speed was selected to reduce noise and the moderate level of edge enhancement was performed to improve the image appearance. Both the lookup tables and the gamma correction were linearized, since these were corrected in the display characterization.

The background images were similarly scanned to a size of 1140 by 980 pixels or twelve by ten inches on the display. This sizing rendered the hands to actual life size.

## 4.5 Colorimetric Image Manipulation

Images were manipulated by each transfer function with the appropriate parameters set to different levels. Four of the six transfer functions were dual sided and symmetry could not be assumed; therefore different parameter levels were used for each side. The exact number of levels for each function was determined from a pilot experiment in order to obtain the optimum perceptibility and acceptability data. These results combined to make sixty-nine parameter levels per image. This did not include the level representing blank trials. Instead of presenting ten blank trials for each image, it was decided to only present two for each image. The final parameter levels for each function are shown in Appendix B.

A pilot experiment was performed to establish perceptibility and

acceptability tolerance ranges. Two images were manipulated using the ten transfer functions, at seven different parameter levels. Twelve color normal observers participated in the pilot experiment. Details of the pilot experiment are discussed in Appendix C.

Once the transfer function parameter levels were established, the actual experimental images were created. The digitized images were transformed into CIELAB lightness, chroma and hue angle dimensions and the appropriate colorimetric manipulations were performed. After each image was manipulated, it was transformed into a displayable format for the Pixar image computer and the calibrated monitor.

Each image was judged for both perceptible and acceptable differences. A total of 852 judgements (426 images) per observer were made. A limit of three one-hour long observation sessions per observer was set to avoid exhaustion of the observers from becoming a factor. The three-hour time limit required that the average pair of perceptibility and acceptability judgements be made in under twenty seconds. The standard and sample images were loaded and displayed in about eight seconds; thus leaving an average of twelve seconds for each pair of judgements. It was later discovered that the loading time could be decreased by a factor of three by eliminating the ethernet data transfer and replacing the optical storage drive with an SMD drive. The storage size of each image was approximately 658 kilobytes, bringing the total disk storage for the experiment to 280 megabytes.

## 4.6 Image Display

A Sony 19 inch trinitron color monitor controlled by the Pixar image computer was calibrated using the following technique. An LMT C1200 colorimeter[92] was used for all tristimulus measurements. The measurement

tolerances in tristimulus coordinates were +/- 0.01. First, the white point was set to the appropriate D65 tristimulus values[4] (X = 95.017, Y = 100.0, and Z = 108.813) using the monitor's internal adjustments with the red, green and blue digital counts maximized. The luminance was set to 85.0 candelas per square meter. Next, the tristimulus values were recorded while decreasing the digital counts of each color channel independently. For example, the first reading would have red channel maximized, with the green and blue channels minimized. Twenty-one levels were recorded for each channel, starting with 2047 and then decrementing down to 0 in even exponential increments. The Pixar digital count range is 2048, while most other computers have a range of 256. These increments were calculated using a power factor of 2.3 (note : the 0 value was not used in the regression model due to negative offsets). A settling time of approximately two seconds was used in the measurements. A non-linear regression model was used for each channel to obtain the appropriate parameters to transform the nonlinear digital counts into linear red, green, and blue values.[93] To convert from tristimulus values to linear red, green, and blue values, a 3x3 matrix was derived by normalizing tristimulus readings for the white point. The readings were normalized against the Y values so that the white point digital counts produce relative tristimulus values equal to D65 values. This matrix was inverted for the inverse transformation. The code used to calibrate the monitor is in Appendix D

The experimental setup included an observer positioned approximately 18 to 24 inches from the display. The monitor displayed background images of two hands on a neutral field holding a 5" by 7" white card. Centered within the card, the test images were displayed in a 4" by 6" field (a similar setup is used for horizontal images). Thus, the observer "saw"

two hands holding a picture with a white border on a grey background as simulated in image 9.

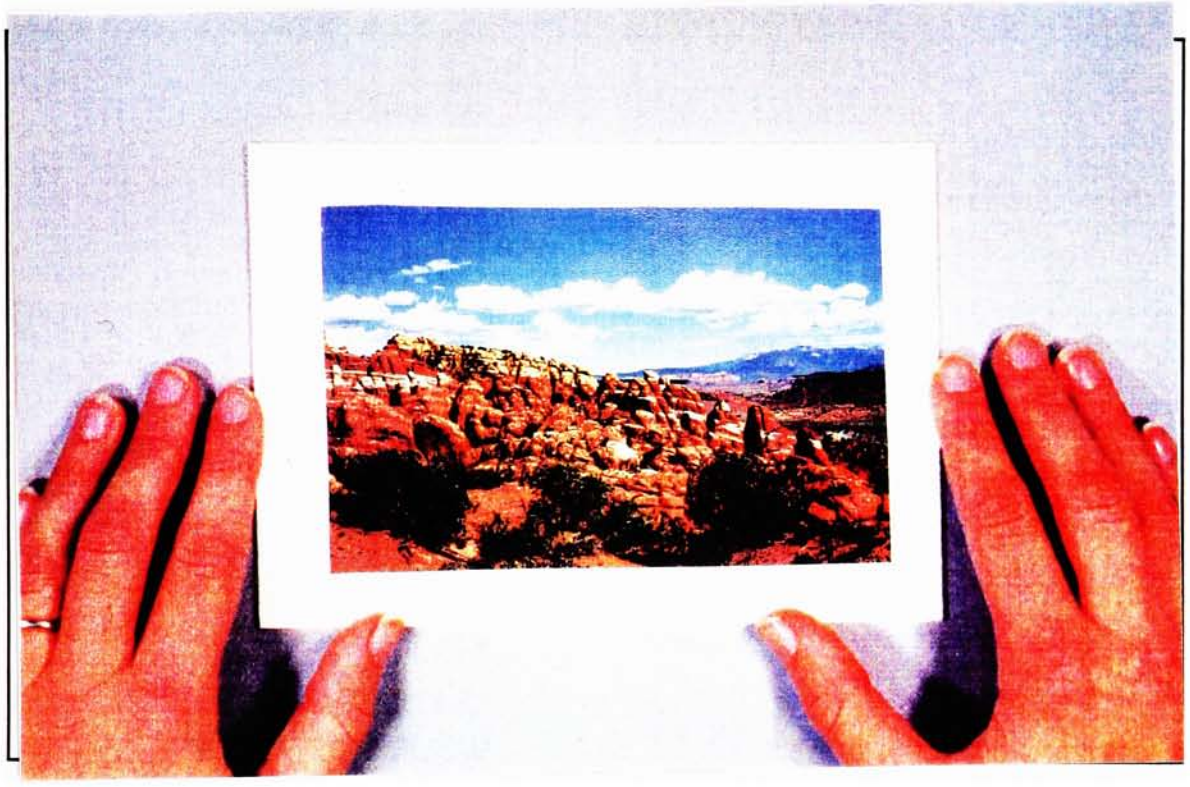


Image 9 : Simulated Observational Image

In the currently implemented paired comparison mode, a calibrated color monitor controlled by the Pixar image computer displayed a choice of two images overlaid on a background. The limitation of two images is imposed by the size of the images, the background and the frame buffer. A separate image is designated as a background in order to provide cognitive effects. The experimental images were placed within the white area to appear more like reflection prints, as opposed to self-luminous images, by invoking the cognitive effects from the background and providing a constant adapting stimulus. The cognitive effect from the background enabled the observer to make judgements closer to reflective prints than self-luminous displays.[94]

This provides a better correlation to most color reproduction systems. The sequential overlaying of two experimental images in the same location was chosen to eliminate the effects of monitor non-uniformity across the screen.

## 4.7 Psychophysical Observations

Thirty-two color normal observers with varied color analysis experience participated in the experiment. The observers varied in age from 20 to 49 and were tested for color vision using a visual colorimeter.[95] A survey was made after the experiment to determine any problems and suggestions for the overall experiment. The results are discussed in Appendix E. The room was darkened during the entire experiment and an approximate temperature of 74 to 78 degrees was maintained.

A neutral field was displayed while the image files were being loaded into display memory. When both images were in memory, the standard image was displayed. Three keys allowed the observer to switch between the reference image, a standard image (which was identical to the reference image) and a manipulated image. In order to preserve the observer's adaptation while viewing actual images, the same grey adaptation field as above was presented for 0.2 seconds when alternating images. The observer stopped on the image that appeared different and then judged if this was an acceptable reproduction or not. After both judgements were made, the next pair of images was displayed.

The following instructions were read to each observer.

> In this experiment, you will be comparing three images at a time. A reference image will always be displayed first, the left button will recall this image at any time. The middle and right buttons toggle between an image that is identical to this reference image and one that has be manipulated in color. Switch between the images, STOPPING on the one that appears different from the reference. If they appear the same, you must still make a choice. If you can't decide, just guess. Once you have decided which image is different, you must decide if this is an acceptable difference or not. Press 'A' if the difference is acceptable and 'N' is it is not acceptable. For this experiment, we are defining acceptability to be 'a reproduction print that you would be expect to find in an expensive book of photographic reproductions.' Many pairs of images will appear identical, please do not let this frustrate you. You should make overall judgements and not compare very small image areas. There are a total of 426 image pairs. There is a five second delay between the images, and a bell will sound when the next image is ready, (please do not press any buttons between images). We will begin with six demonstration images to make sure you understand the directions.

One of the difficulties encountered was obtaining both perceptibility and acceptability tolerances in a single experiment. Although the instructions were lengthy, most observers perceived no difficulties. Six practice trials were given in order to eliminate any instructional misunderstandings.

When measuring acceptability tolerances, an acceptability context must be established. This condition limits acceptability tolerances to particular circumstances such as "acceptable automotive paint match or textile match." The acceptability context was created with the cooperation of R. R. Donnelley and Sons Company. This context was derived from a hardcopy prepress proofing environment.

## 4.8 Statistical Evaluation

The experimental results were blocked by each original image. The observational data were analyzed for goodness of fit, actual perceptibility and acceptability tolerances and uncertainty estimates for these tolerances. The end results were compared against common color difference formulas. Probit analysis was used to analyze the data. Probit analysis is a maximum likelihood model relating the experimental responses to occurrence probability estimates. This model fits the frequency of cumulative observer responses to a cumulative normal distribution. The Pearson Chi-Squared test[96] and its associated probability determined how well the data fit the cumulative normal distribution assumed in the probit analysis. Estimation of tolerances and uncertainty of these tolerances are derived from this model. The most accurate estimate is the median tolerance (T50) at a rejection/acceptance probability of 50%. The fiducial limits (95% confidence limits denoted LOWER and UPPER) were calculated to produce an estimate of uncertainty for the T50 results. Fiducial limits are expressions of the probability that within a certain percentage, the estimate will fall in a particular range. When the model fit is very poor, the fiducial limits are infinite in value. This condition is noted by a "." in the data below. Finney,[96] Alman,[97] and Berns[9] have provided very detailed presentations of probit analysis and fiducial limits.

The SAS Logistic procedure was used to determine the Pearson Chi-Squared values and probabilities and the C statistic. The C statistic is a measurement of parameter sensitivity and thus model fit.[98] These C statistics were used to isolate and eliminate the extreme five percent of the observations. The tolerance mean and fiducial limits were derived using the

SAS probit analysis procedure. The two separate SAS procedures were used since only the logit procedure provided discrimination parameters and only the probit procedure provided fiducial limits.

# 5. Results and Discussion

## 5.1 Statistical Significance

Visual experiments are notorious for containing a lot of noise in their results. Previous visual experiments have shown that goodness of fit Chi-Squared probabilities of five percent or greater yield very sound results.[99]

The Chi-Squared values and their related probabilities for this experiment are shown in table 5.1-1. Several transfer functions indicate noise with their Chi-Squared probabilities being less than 0.05. Two such examples are the low side of the chroma power function (CPL) for both the perceptibility and acceptability data.

| Raw | Perceptibility | | Acceptability | |
|---|---|---|---|---|
| Function | Chi $^2$ | Pr > Chi $^2$ | Chi $^2$ | Pr > Chi $^2$ |
| LMF | 11.16 | 0.02 | 12.37 | 0.01 |
| LPH | 2.61 | 0.62 | 3.68 | 0.45 |
| LPL | 5.67 | 0.23 | 10.12 | 0.04 |
| LSH | 61.93 | 0.00 | 32.44 | 0.00 |
| LSL | 11.45 | 0.02 | 8.14 | 0.09 |
| CMF | 5.63 | 0.23 | 14.04 | 0.01 |
| CPH | 14.38 | 0.03 | 7.27 | 0.30 |
| CPL | 49.10 | 0.00 | 24.44 | 0.00 |
| HOH | 4.62 | 0.46 | 21.21 | 0.00 |
| HOL | 39.30 | 0.00 | 66.09 | 0.00 |

Table 5.1-1 : Raw Chi-Squared Statistics

Previous researchers have filtered data to reduce some of the noise and improve the statistical significance.[9] Such filtering was justified in this experiment by an estimated "key stroke error rate" of four percent. This error rate was derived from a one percent rate of "fail/standard" response combinations. This response would indicate that the observer judged the

standard image to be an unacceptable reproduction of the reference image. This is obviously wrong, since these images were identical, so it was assumed that the observer miskeyed the response. Since this is one of four possible combinations, a total error rate of four percent was estimated. This error rate justifies filtering out up to four percent of the data.

The SAS Logistic procedure provides seven regression diagnostic statistics based on work by Pregibon.[100] These diagnostic statistics were developed to identify influential observations. Each of these diagnostic statistics were tested as a possible filter criterion by using the experimental results and seeking the lowest Chi-Squared values. The C diagnostic statistic was most effective and was used to filter out one and one-half percent of the total data. This particular statistic corresponds to observations that have undue influence on both individual coefficients and the model fit itself. The comparison of raw and filtered results are shown in Tables 5.1-2 and 5.1-3. The perceptibility results were significantly improved by this filtering technique and strong statistical significance with a very low noise level is achieved. To give some perspective of the level of these results, they are an improvement by a factor of two over previous MCSL results[9] and by a factor of four over Luo and Rigg results.[101] The acceptability tolerances were not significantly affected by filtering, indicating that a substantial amount of observer noise is indeed in the results. This was not unexpected. In the post-experimental survey, several observers stated that they ignored the acceptability criteria and used their own criteria for most of the experiment. Such behavior would create multiple acceptance tolerances and thus heterogeneous noise in the acceptability data depending on each observer's acceptability criteria. The individual Chi-Squared statistics shown in Table 5.1-4, support this argument by showing less noise for the individual results

than the grouped results. Since the acceptability data was not improved by filtering the raw results were used in further analysis. Therefore, only the perceptibility data was filtered by eliminating a total of one and one-half percent of the data.

Table 5.1-2 : Perceptibility Chi-Squared Statistics

| | Raw Perceptibility | | Filtered Perceptibility | |
|---|---|---|---|---|
| Function | Chi $^2$ | Pr > Chi $^2$ | Chi $^2$ | Pr > Chi $^2$ |
| LMF | 11.16 | 0.02 | 7.95 | 0.09 |
| LPH | 2.61 | 0.62 | 2.98 | 0.56 |
| LPL | 5.67 | 0.23 | 5.50 | 0.24 |
| LSH | 61.93 | 0.00 | 8.52 | 0.38 |
| LSL | 11.45 | 0.02 | 7.41 | 0.12 |
| CMF | 5.63 | 0.23 | 5.63 | 0.23 |
| CPH | 14.38 | 0.03 | 10.83 | 0.09 |
| CPL | 49.10 | 0.00 | 4.96 | 0.42 |
| HOH | 4.62 | 0.46 | 5.47 | 0.36 |
| HOL | 39.30 | 0.00 | 4.39 | 0.49 |

Table 5.1-3 : Acceptability Chi-Squared Statistics

| | Raw Acceptability | | Filtered Acceptability | |
|---|---|---|---|---|
| Function | Chi $^2$ | Pr > Chi $^2$ | Chi $^2$ | Pr > Chi $^2$ |
| LMF | 12.37 | 0.01 | 12.37 | 0.01 |
| LPH | 3.68 | 0.45 | 3.68 | 0.45 |
| LPL | 10.12 | 0.04 | 10.12 | 0.04 |
| LSH | 32.44 | 0.00 | 32.44 | 0.00 |
| LSL | 8.14 | 0.09 | 8.14 | 0.09 |
| CMF | 14.04 | 0.01 | 7.44 | 0.11 |
| CPH | 7.27 | 0.30 | 7.27 | 0.30 |
| CPL | 24.44 | 0.00 | 14.31 | 0.01 |
| HOH | 21.21 | 0.00 | 21.21 | 0.00 |
| HOL | 66.09 | 0.00 | 30.44 | 0.00 |

The individual observer's Chi-Squared statistics show that each observer was self consistent and displayed little variation between images. This is seen by the very high probabilities and low Chi-Squared values in comparison with the values in tables 5.1-2 and 5.1-3.

Table 5.1-4 : Individual Observer's Chi-Squared Statistics

| Observer | Perceptibility | | Acceptability | |
|---|---|---|---|---|
| | Chi $^2$ | Pr > Chi $^2$ | Chi $^2$ | Pr > Chi $^2$ |
| adn | 2.04 | 0.86 | 4.19 | 0.66 |
| bas | 3.50 | 0.78 | 3.61 | 0.71 |
| bdn | 3.85 | 0.70 | 2.61 | 0.83 |
| bds | 6.04 | 0.45 | 3.12 | 0.78 |
| bjr | 7.59 | 0.34 | 3.68 | 0.71 |
| cjw | 3.08 | 0.77 | 3.98 | 0.69 |
| cmm | 3.06 | 0.76 | 4.41 | 0.64 |
| cmr | 2.56 | 0.82 | 3.11 | 0.78 |
| eap | 3.31 | 0.75 | 4.76 | 0.60 |
| emh | 3.45 | 0.74 | 3.22 | 0.73 |
| gcm | 4.41 | 0.62 | 5.78 | 0.50 |
| jmm | 3.49 | 0.73 | 4.21 | 0.64 |
| jtn | 1.59 | 0.93 | 2.46 | 0.84 |
| jwp | 2.53 | 0.82 | 2.63 | 0.82 |
| jxr | 4.65 | 0.60 | 6.50 | 0.47 |
| kap | 3.39 | 0.73 | 5.76 | 0.66 |
| ksm | 0.96 | 0.97 | 2.50 | 0.81 |
| lar | 1.44 | 0.94 | 2.35 | 0.86 |
| lxw | 4.77 | 0.57 | 3.02 | 0.77 |
| mdf | 2.65 | 0.82 | 3.13 | 0.78 |
| meg | 5.41 | 0.52 | 4.68 | 0.64 |
| mkm | 4.54 | 0.61 | 4.86 | 0.55 |
| mxm | 4.17 | 0.66 | 3.68 | 0.72 |
| pat | 3.20 | 0.74 | 5.02 | 0.64 |
| pxh | 1.13 | 0.97 | 1.34 | 0.95 |
| rjl | 6.43 | 0.45 | 3.65 | 0.71 |
| rsb | 4.96 | 0.60 | 2.73 | 0.79 |
| sls | 3.23 | 0.74 | 3.68 | 0.71 |
| sma | 2.98 | 0.79 | 4.78 | 0.63 |
| sms | 5.08 | 0.55 | 4.84 | 0.68 |

| | | | | |
|---|---|---|---|---|
| trh | 3.14 | 0.76 | 3.29 | 0.75 |
| yxl | 3.15 | 0.77 | 5.76 | 0.66 |

## 5.2 Perceptibility Tolerances

The experimental perceptibility tolerances are shown in table 5.2-1. Notable are the tolerance values themselves, the tight fiducial limits, the apparent symmetry for all of the dual sided functions and the hue-angle offset of five degrees. These tolerance values can be used to perceptually discriminate between original and reproduction images. The tight fiducial limits allow the results to be useful for tolerances. The symmetry can be used in future experiments to eliminate many of the transfer functions as redundant, thus either shortening the observational times or allowing for a large number of images to be judged. The hue angle of five degrees was a surprise because previous work indicated that observers were extremely sensitive to hue angle shifts. A five degree shift indicates that observers are much less sensitive to shifts in hue angle of images than uniform fields.

Table 5.2-1 : Perceptibility Results

| Function | T50 | LOWER | UPPER | Pr > Chi $^2$ |
|---|---|---|---|---|
| LMF | 0.93 | 0.92 | 0.95 | 0.09 |
| LPH | 1.11 | 1.10 | 1.13 | 0.56 |
| LPL | 0.90 | 0.89 | 0.92 | 0.24 |
| LSH | 1.17 | 1.16 | 1.19 | 0.38 |
| LSL | 0.88 | 0.87 | 0.89 | 0.12 |
| CMF | 0.91 | 0.90 | 0.92 | 0.23 |
| CPH | 1.13 | 1.11 | 1.14 | 0.09 |
| CPL | 0.89 | 0.88 | 0.9 | 0.42 |
| HOH | 5.9 | 5.2 | 6.5 | 0.36 |
| HOL | -4.9 | -5.6 | -4.1 | 0.49 |

## 5.3 Acceptability Tolerances

The experimental acceptability tolerances are shown in table 5.3-1. Again notable are the tolerances themselves, the tight fiducial limits and the symmetry for all of the dual sided functions. These tolerance values can be used to acceptably discriminate between original and reproduction images. The acceptability tolerances are greater than the perceptibility tolerances by a factor of 1.59 with a standard deviation of 0.21. Again, the symmetry can be used in future experiments to eliminate many of the transfer functions as redundant. Most importantly, the Chi-Squared probabilities indicate a poor model fit of the cumulative normal distribution for many of the transfer functions despite the tight fiducial tolerances. Since the probilities for individual scenes indicate an adequate model fit, it seems reasonable to attribute this random noise due to different observer criteria. This is supported in the survey comments as shown in Appendix E.

Table 5.3-1 : Acceptability Results

| Function | T50 | LOWER | UPPER | Pr > Chi $^2$ |
|---|---|---|---|---|
| LMF | 0.91 | 0.90 | 0.91 | 0.01 |
| LPH | 1.19 | 1.18 | 1.20 | 0.45 |
| LPL | 0.87 | 0.85 | 0.88 | 0.04 |
| LSH | 1.35 | 1.32 | 1.38 | 0.00 |
| LSL | 0.84 | 0.82 | 0.85 | 0.09 |
| CMF | 0.85 | 0.83 | 0.87 | 0.01 |
| CPH | 1.18 | 1.17 | 1.18 | 0.30 |
| CPL | 0.80 | 0.77 | 0.82 | 0.00 |
| HOH | 9.2 | 8.2 | 10.3 | 0.00 |
| HOL | -8.9 | -11.1 | -6.3 | 0.00 |

## 5.4 Perceptibility Tolerances by Scene

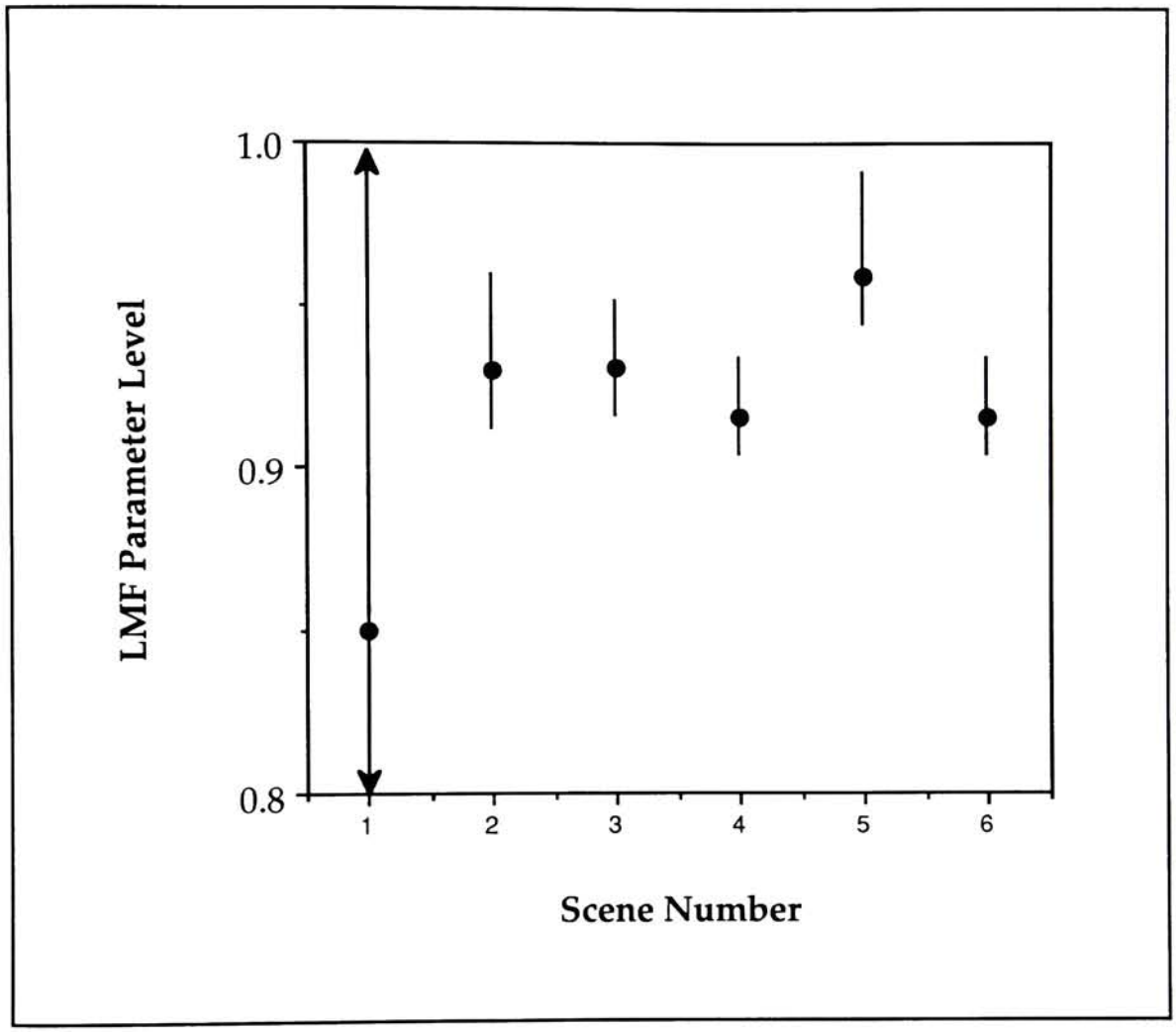
Table 5.4-1 and graphs 5.4-1 through 5.4-10 show the results broken down by scene numerically and graphically. If no differences between scenes existed, a horizontal line could connect all six of the image results in these graphs. While this is not the case, most of the images not intercepted by this line are not significantly visually different. The exceptions are images without mid-chromatic tones are more sensitive to increases in chroma and images which are within a relatively small hue angle arc are not as sensitive to negative hue shifts. This is notable in that some standard color checkers are also without mid-chromatic tones and thus possibly do not simulate image differences well. Images with little mid-chromatic tones such as scene #6 and images within a relatively small hue angle are such as scene #5 stand out in graphs 5.4-7 and 5.4-10. All of the other transfer functions indicate no significant visual difference between scenes.

Table 5.4-1 : Perceptibility Results by Scene

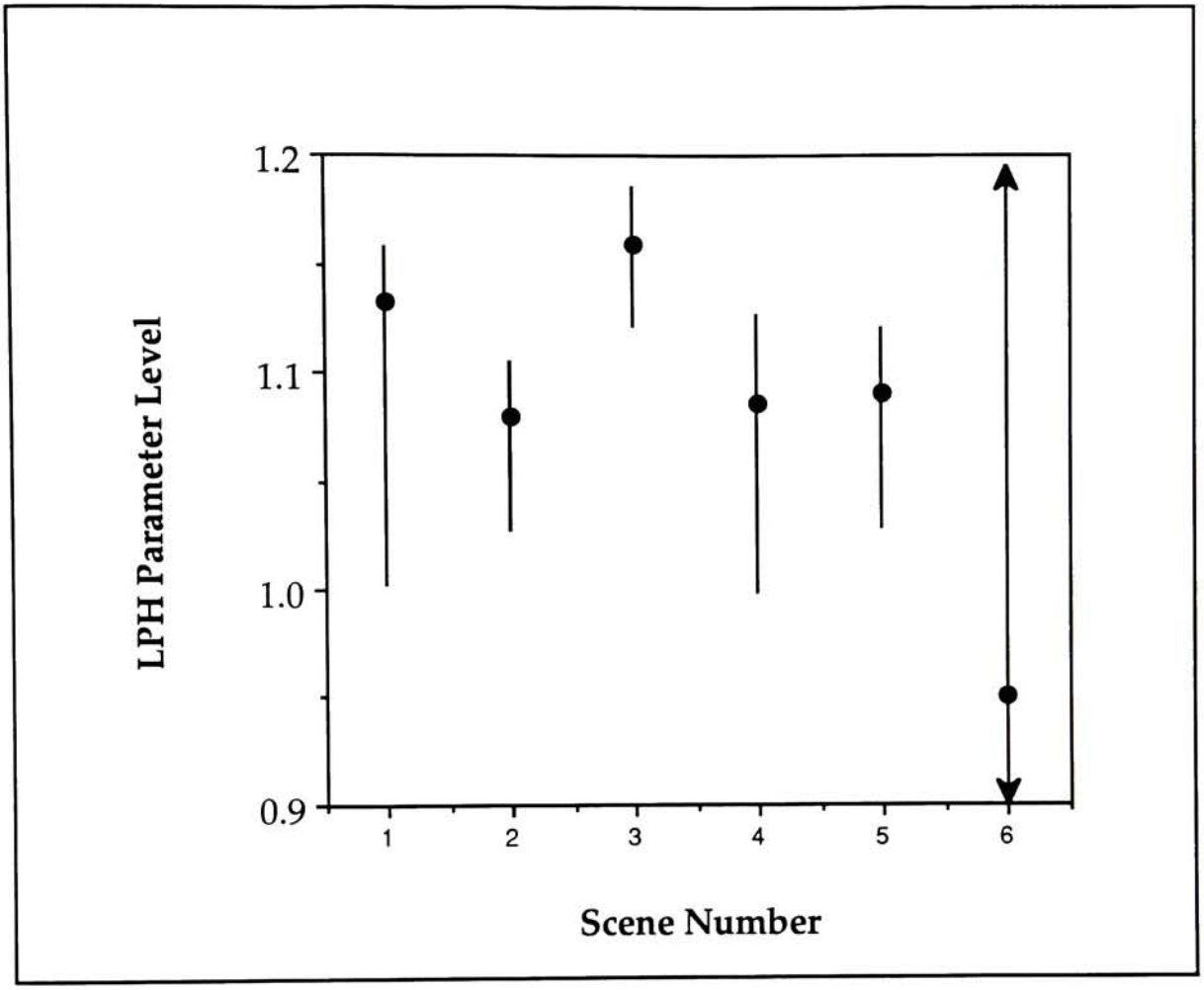
| Scene | # | Function | T50 | LOWER | UPPER | Prob > Chi $^2$ |
|---|---|---|---|---|---|---|
| gcmpnl | 1 | LMF | 0.94 | . | . | 0.01 |
| mdfnfh | 2 | LMF | 0.93 | 0.91 | 0.96 | 0.43 |
| q60mfh | 3 | LMF | 0.93 | 0.92 | 0.95 | 0.47 |
| q60pfh | 4 | LMF | 0.92 | 0.90 | 0.93 | 0.19 |
| rcimnl | 5 | LMF | 0.96 | 0.94 | 0.99 | 0.43 |
| smannl | 6 | LMF | 0.92 | 0.90 | 0.93 | 0.44 |
| gcmpnl | 1 | LPH | 1.13 | 1.00 | 1.16 | 0.92 |
| mdfnfh | 2 | LPH | 1.08 | 1.03 | 1.10 | 0.73 |
| q60mfh | 3 | LPH | 1.16 | 1.12 | 1.19 | 0.21 |
| q60pfh | 4 | LPH | 1.09 | 1.00 | 1.13 | 0.60 |
| rcimnl | 5 | LPH | 1.09 | 1.03 | 1.12 | 0.65 |
| smannl | 6 | LPH | 1.12 | . | . | 0.01 |
| gcmpnl | 1 | LPL | 0.90 | 0.88 | 0.94 | 0.91 |
| mdfnfh | 2 | LPL | 0.96 | 0.92 | 1.08 | 0.34 |

| | | | | | | |
|---|---|---|---|---|---|---|
| q60mfh | 3 | LPL | 0.89 | 0.87 | 0.94 | 0.77 |
| q60pfh | 4 | LPL | 0.90 | 0.88 | 0.95 | 0.11 |
| rcimnl | 5 | LPL | 0.90 | 0.88 | 0.93 | 0.58 |
| smannl | 6 | LPL | 0.89 | 0.86 | 0.93 | 0.12 |
| gcmpnl | 1 | LSH | 1.17 | 1.13 | 1.21 | 0.23 |
| mdfnfh | 2 | LSH | 1.17 | 1.13 | 1.20 | 0.90 |
| q60mfh | 3 | LSH | 1.20 | 1.06 | 1.28 | 0.01 |
| q60pfh | 4 | LSH | 1.15 | 1.12 | 1.17 | 0.91 |
| rcimnl | 5 | LSH | 1.17 | 1.13 | 1.20 | 0.57 |
| smannl | 6 | LSH | 1.18 | 1.14 | 1.21 | 0.81 |
| gcmpnl | 1 | LSL | 0.88 | 0.82 | 1.01 | 0.05 |
| mdfnfh | 2 | LSL | 0.86 | 0.83 | 0.89 | 0.82 |
| q60mfh | 3 | LSL | 0.84 | 0.81 | 0.88 | 0.21 |
| q60pfh | 4 | LSL | 0.90 | 0.88 | 0.94 | 0.46 |
| rcimnl | 5 | LSL | 0.88 | 0.86 | 0.92 | 0.36 |
| smannl | 6 | LSL | 0.88 | 0.86 | 0.91 | 0.54 |
| gcmpnl | 1 | CMF | 0.92 | 0.89 | 0.99 | 0.84 |
| mdfnfh | 2 | CMF | 0.89 | 0.87 | 0.93 | 0.73 |
| q60mfh | 3 | CMF | 0.90 | 0.84 | 1.03 | 0.10 |
| q60pfh | 4 | CMF | 0.89 | 0.86 | 0.94 | 0.79 |
| rcimnl | 5 | CMF | 0.97 | . | . | 0.01 |
| smannl | 6 | CMF | 0.88 | 0.86 | 0.92 | 0.98 |
| gcmpnl | 1 | CPH | 1.13 | 1.10 | 1.16 | 0.56 |
| mdfnfh | 2 | CPH | 1.15 | 1.08 | 1.20 | 0.03 |
| q60mfh | 3 | CPH | 1.14 | 1.10 | 1.18 | 0.95 |
| q60pfh | 4 | CPH | 1.11 | 1.07 | 1.14 | 0.17 |
| rcimnl | 5 | CPH | 1.07 | 1.02 | 1.09 | 0.75 |
| smannl | 6 | CPH | 1.14 | 1.08 | 1.20 | 0.02 |
| gcmpnl | 1 | CPL | 0.90 | 0.88 | 0.93 | 0.69 |
| mdfnfh | 2 | CPL | 0.85 | 0.76 | 1.02 | 0.06 |
| q60mfh | 3 | CPL | 0.86 | 0.83 | 0.90 | 0.88 |
| q60pfh | 4 | CPL | 0.90 | 0.88 | 0.94 | 0.85 |
| rcimnl | 5 | CPL | 0.90 | 0.87 | 0.93 | 0.93 |
| smannl | 6 | CPL | 0.90 | 0.81 | 1.11 | 0.07 |
| gcmpnl | 1 | HOH | 6.2 | 3.9 | 7.7 | 0.60 |
| mdfnfh | 2 | HOH | 5.2 | 3.8 | 6.3 | 0.63 |
| q60mfh | 3 | HOH | 5.5 | 3.3 | 6.9 | 0.94 |
| q60pfh | 4 | HOH | 7.4 | 5.2 | 8.9 | 0.93 |
| rcimnl | 5 | HOH | 3.2 | . | . | 1.00 |
| smannl | 6 | HOH | 8.8 | 6.8 | 10.2 | 0.50 |
| gcmpnl | 1 | HOL | -3.6 | . | . | 0.00 |
| mdfnfh | 2 | HOL | -4.2 | -5.1 | -2.7 | 0.75 |
| q60mfh | 3 | HOL | -5.2 | -6.6 | -2.7 | 0.86 |

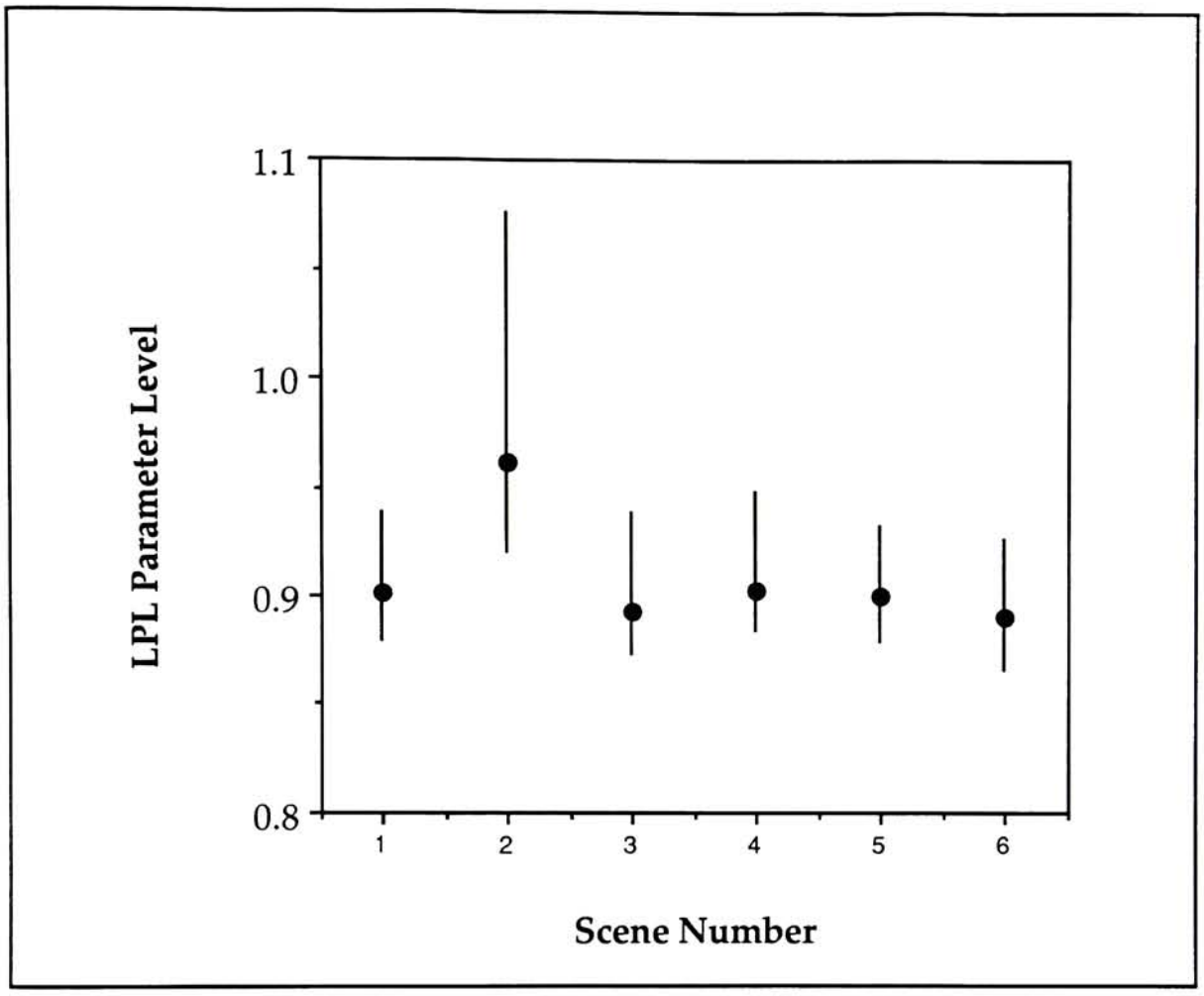
| | | | | | | |
|---|---|---|---|---|---|---|
| q60pfh | 4 | HOL | -5.7 | -7.8 | -1.7 | 0.27 |
| rcimnl | 5 | HOL | -2.7 | -3.9 | 0.3 | 0.76 |
| smannl | 6 | HOL | -7.7 | -9.3 | -5.4 | 0.27 |



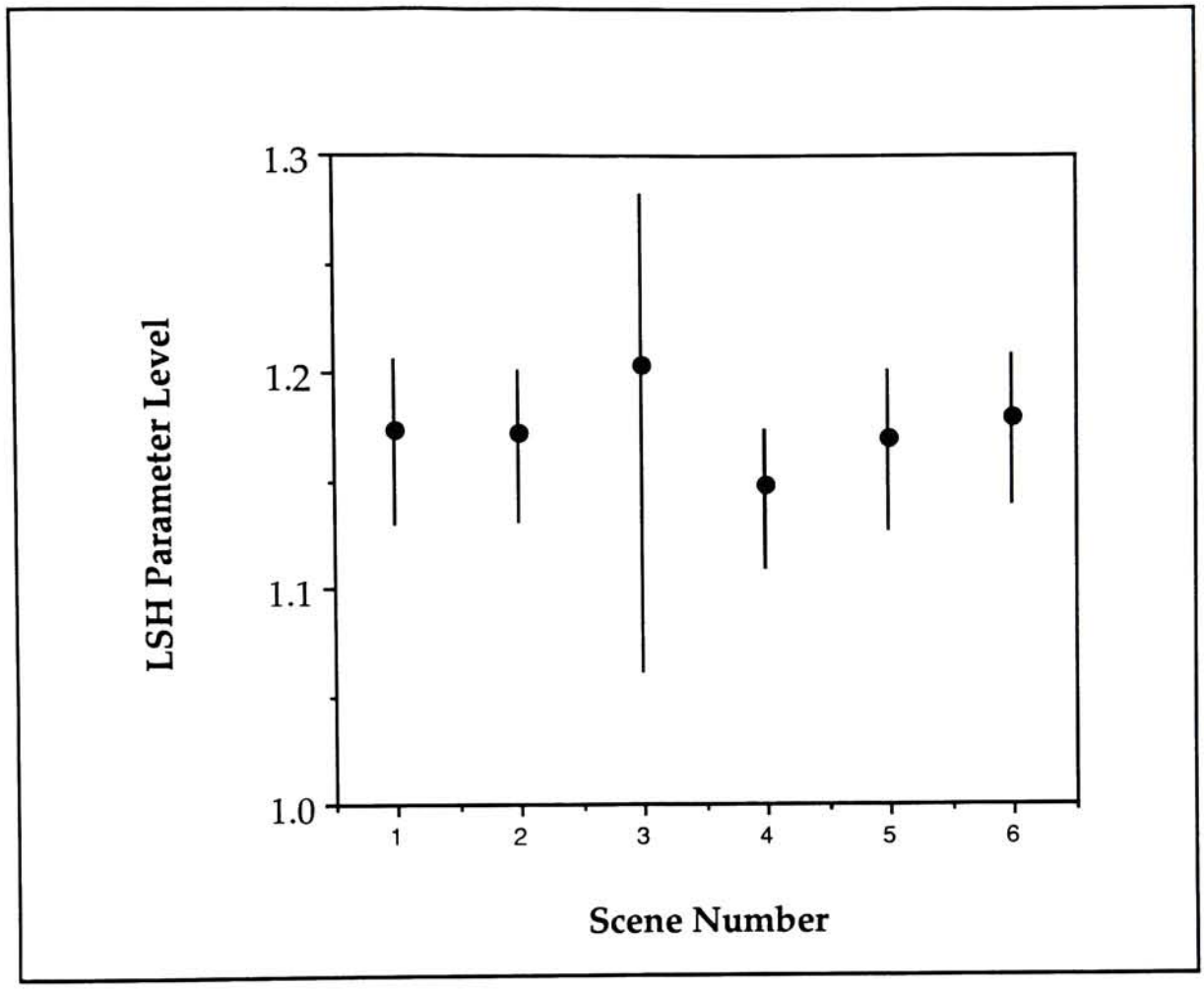

Graph 5.4-1 : Perceptibility by Scene for Lightness Multiplicative Factor



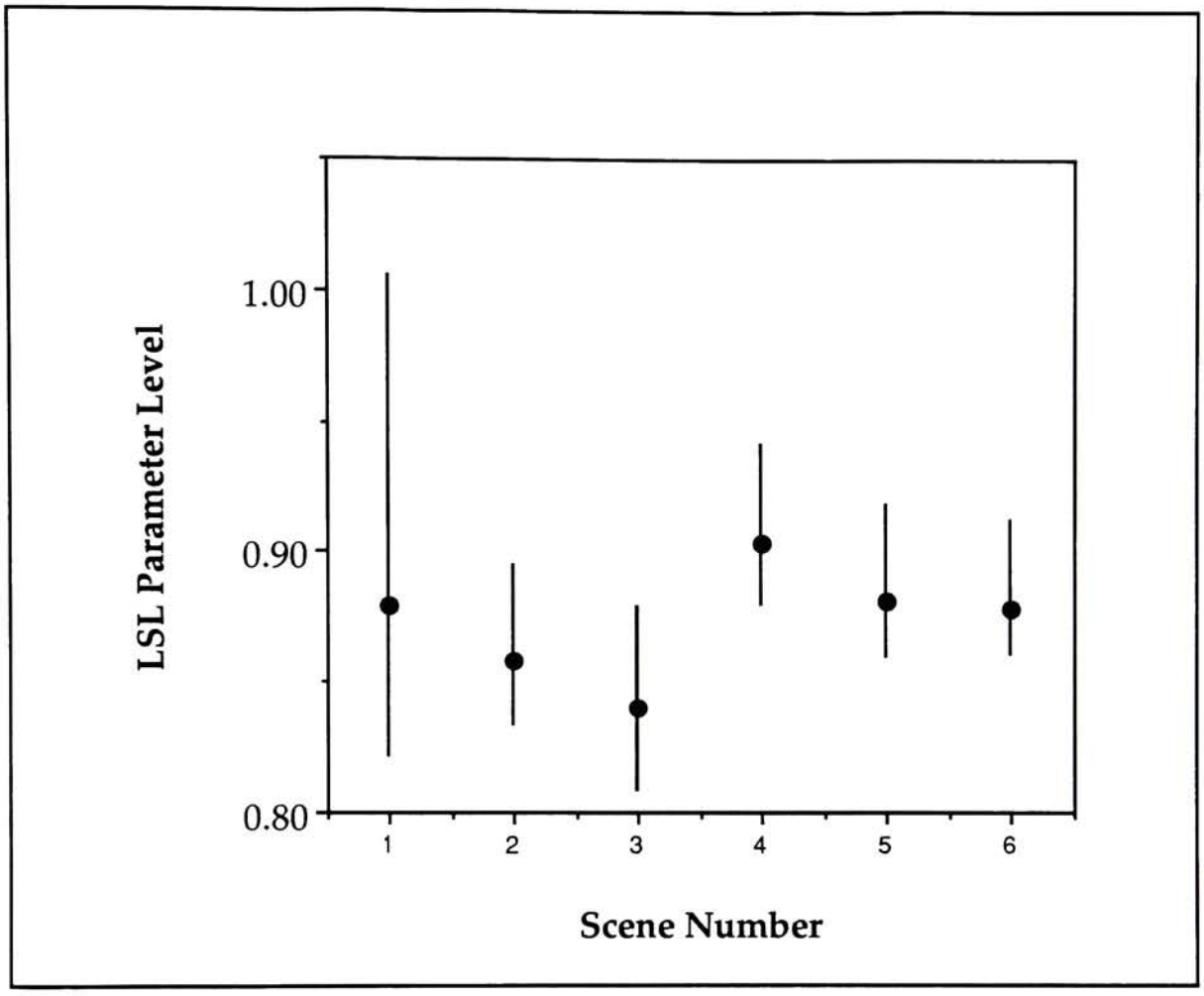

Graph 5.4-2 : Perceptibility by Scene for Lightness High Power



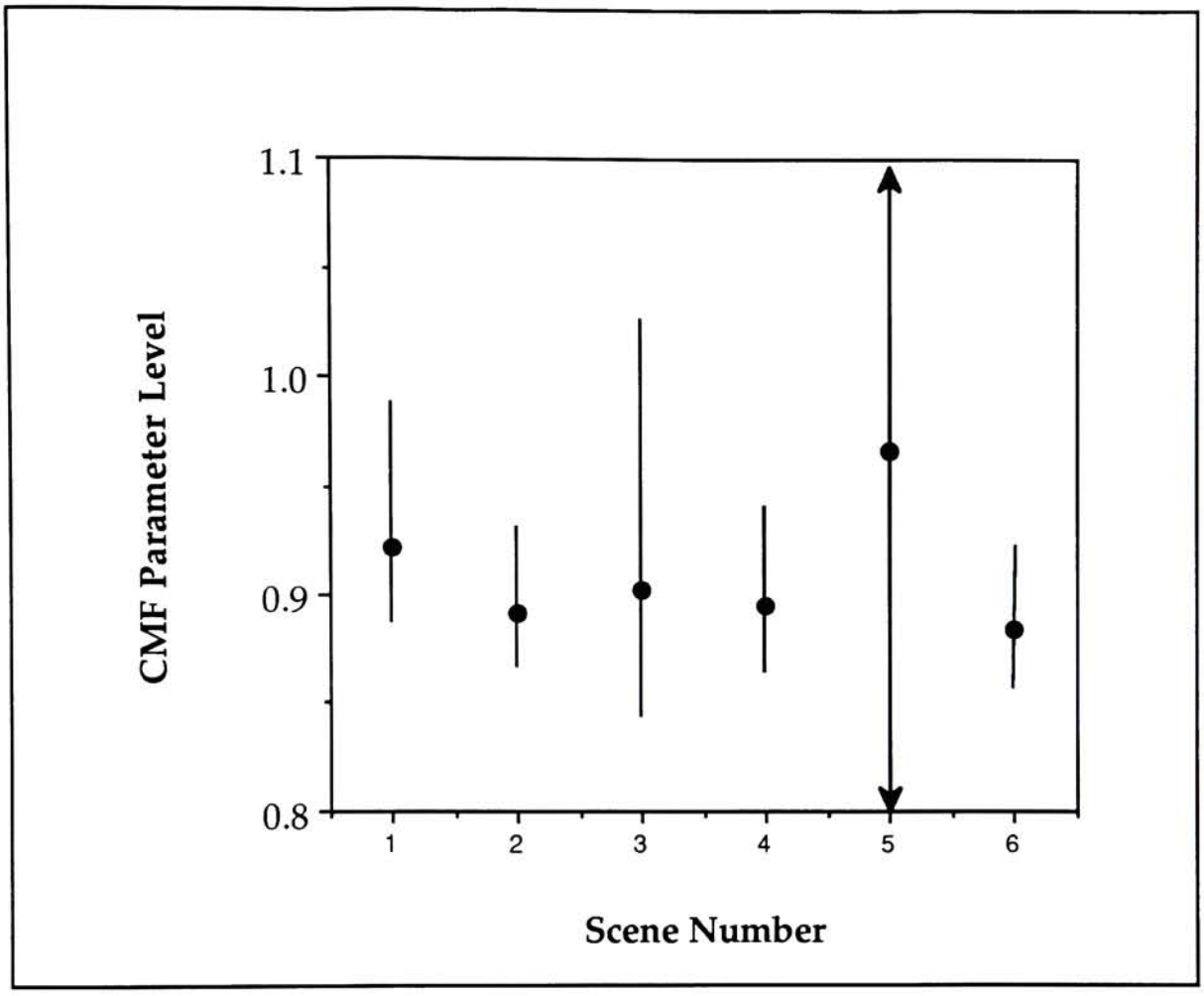

Graph 5.4-3 : Perceptibility by Scene for Lightness Low Power



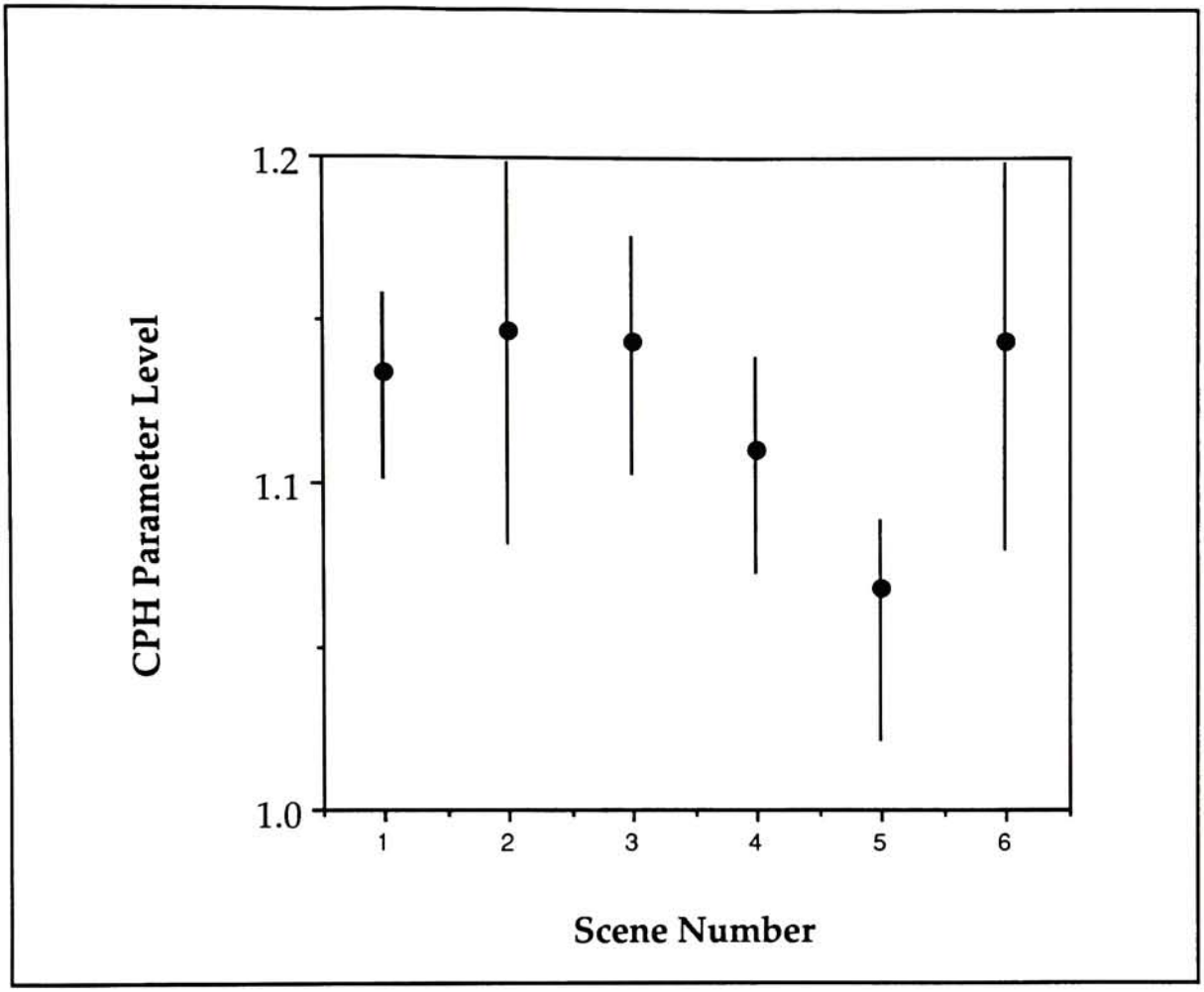

Graph 5.4-4 : Perceptibility by Scene for Lightness High Sigmoidal



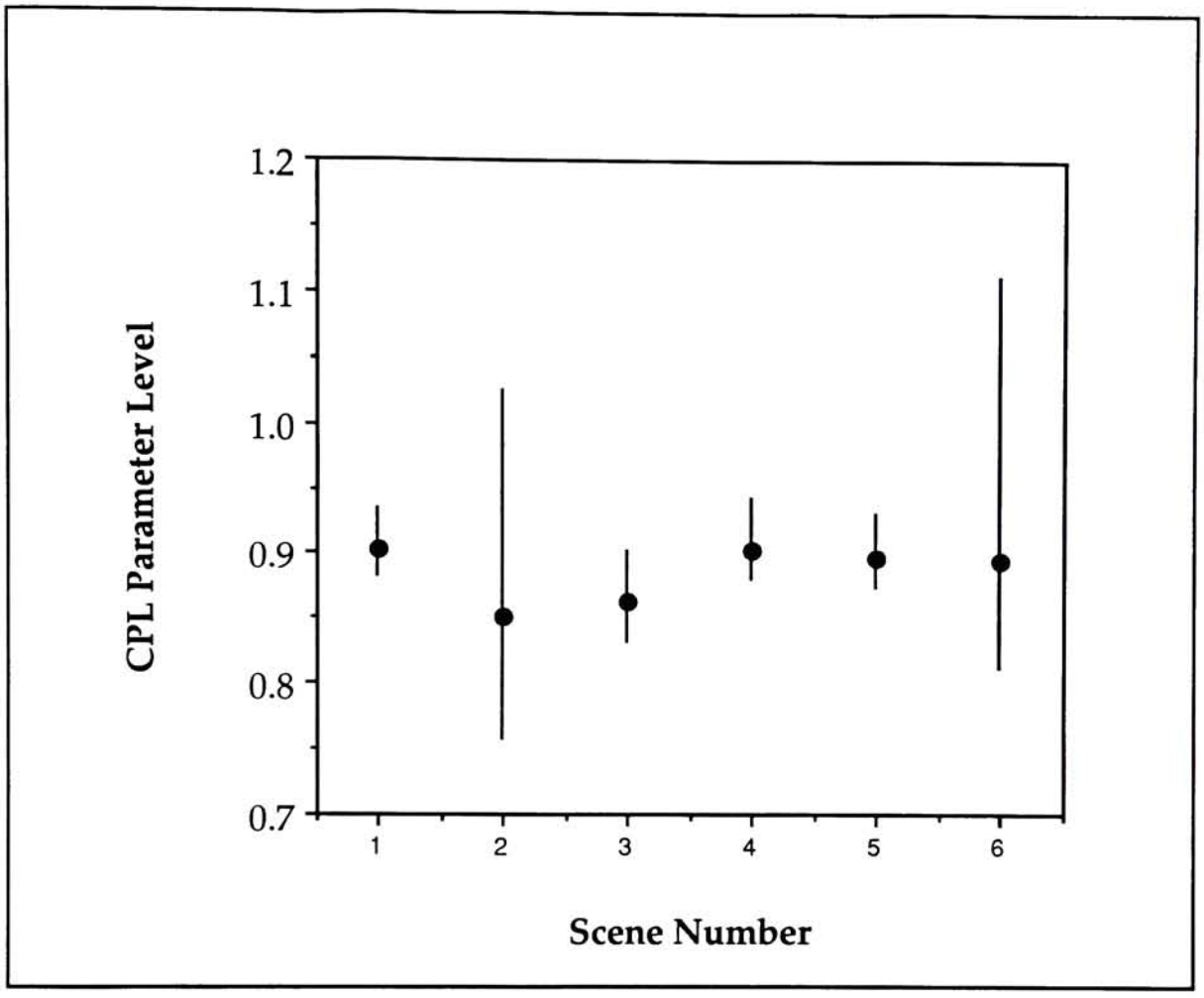

Graph 5.4-5 : Perceptibility by Scene for Lightness Low Sigmoidal



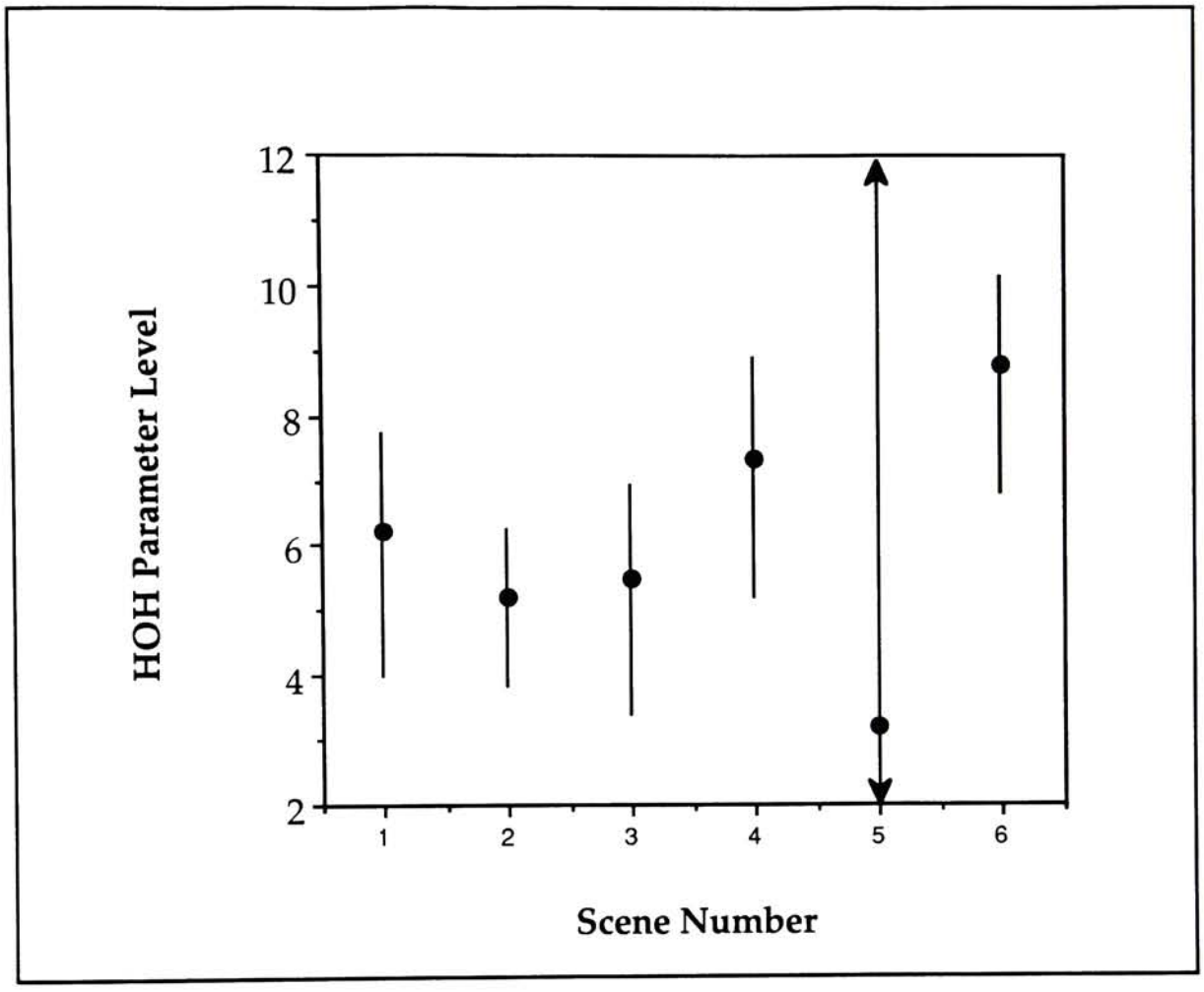

Graph 5.4-6 : Perceptibility by Scene for Chroma Multiplicative Factor



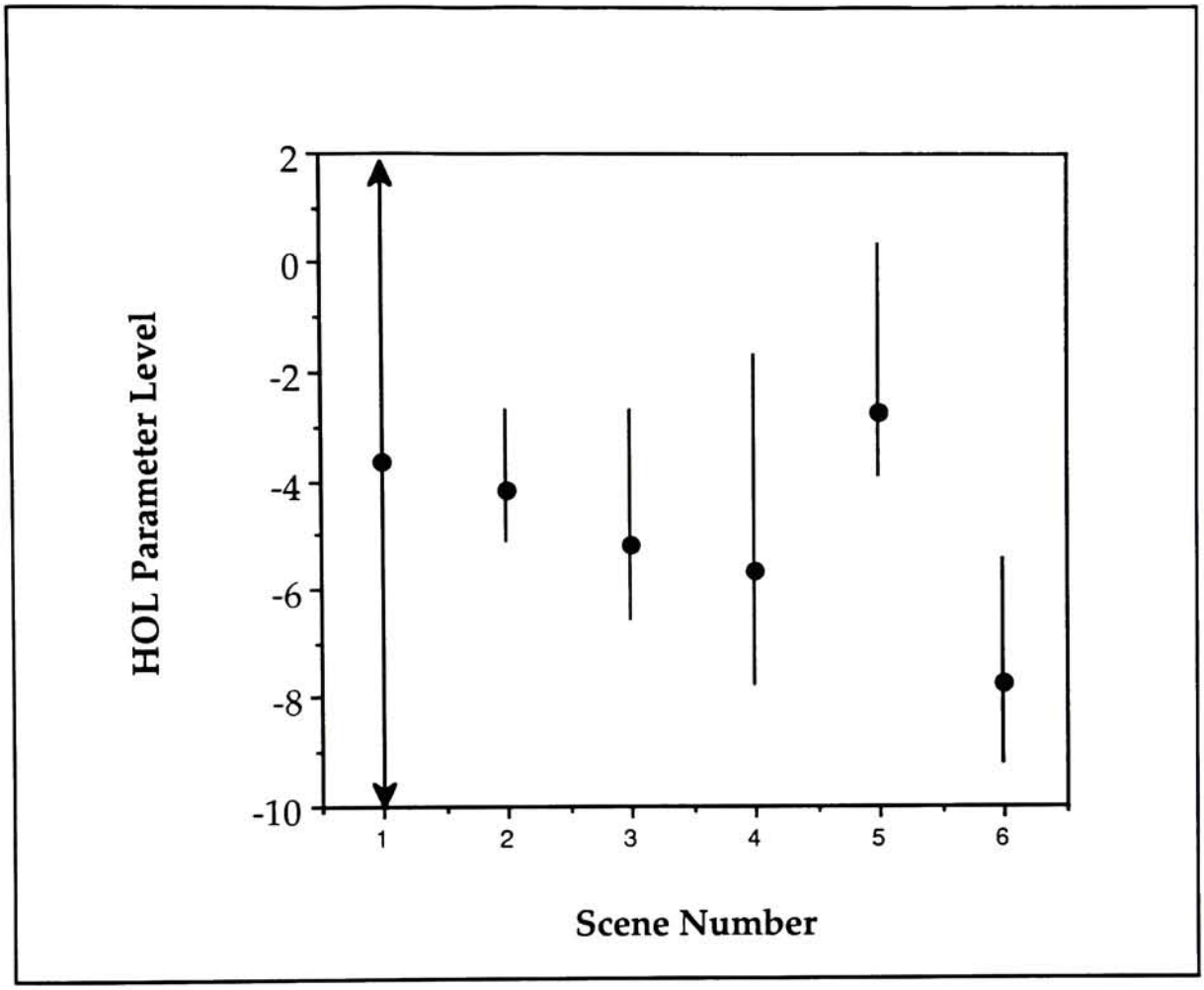

Graph 5.4-7 : Perceptibility by Scene for Chroma High Power




Graph 5.4-8 : Perceptibility by Scene for Chroma Low Power




Graph 5.4-9 : Perceptibility by Scene for Hue Angle Positive Offset




Graph 5.4-10 : Perceptibility by Scene for Hue Angle Negative Offset

## 5.5 Acceptability Tolerances by Scene

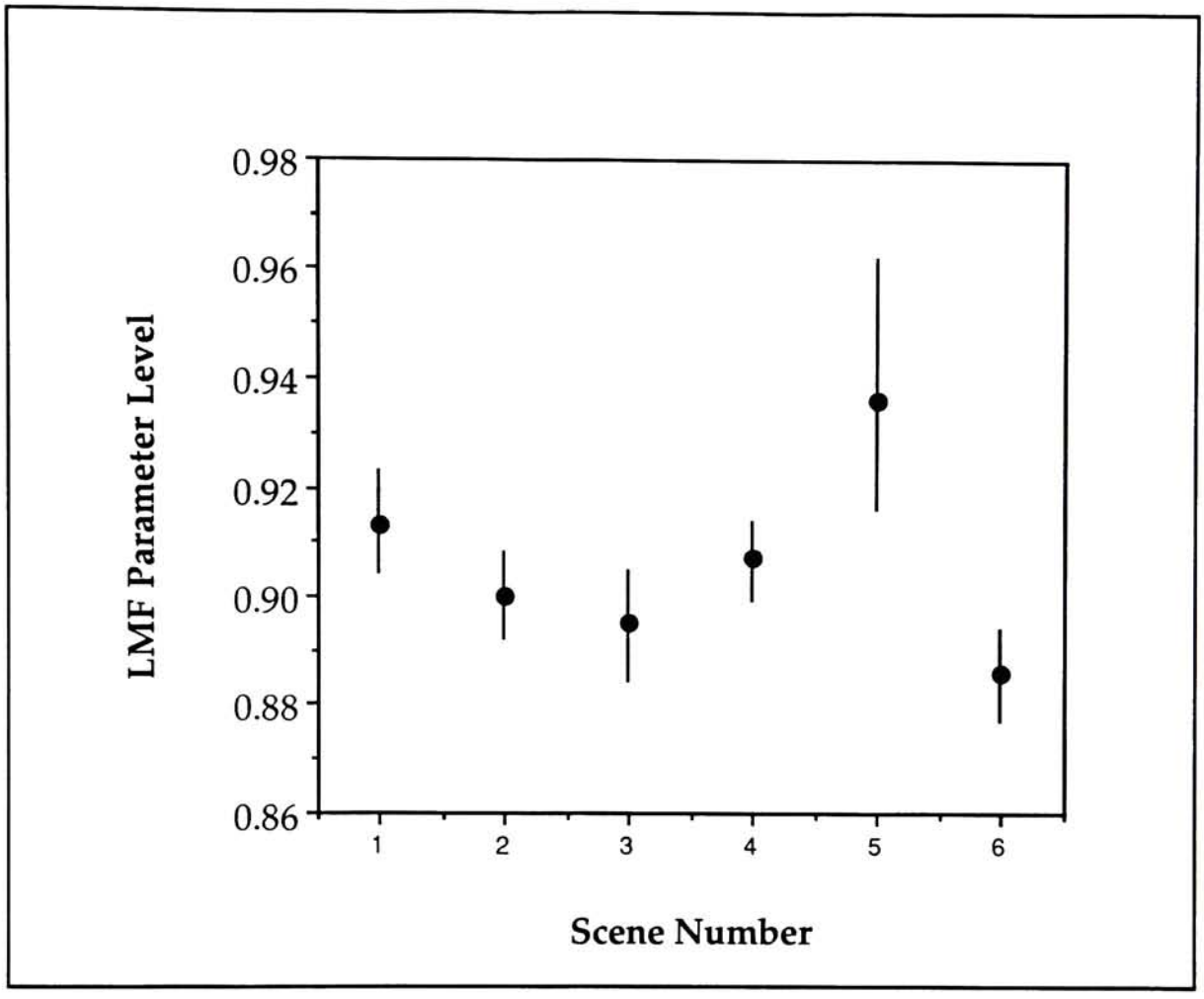
Table 5.5-1 and graphs 5.5-1 through 5.5-10 show the acceptability results broken down by scene numerically and graphically. These results indicate that there exists little significant difference between scenes. The exceptions are images which are within a relatively small hue angle arc are more sensitive to increases in chroma and images without mid-chromatic tones are not as sensitive to negative hue shifts. Again scenes #5 and #6 in graphs 5.5-7 and 5.5-10 are examples of such scenes. All of the other transfer

functions indicate no significant visual difference between scenes. The higher Chi-Squared probabilities (compared with the overall acceptability data) indicate random noise that can be attributed to different observers using different acceptability criteria for different scenes.

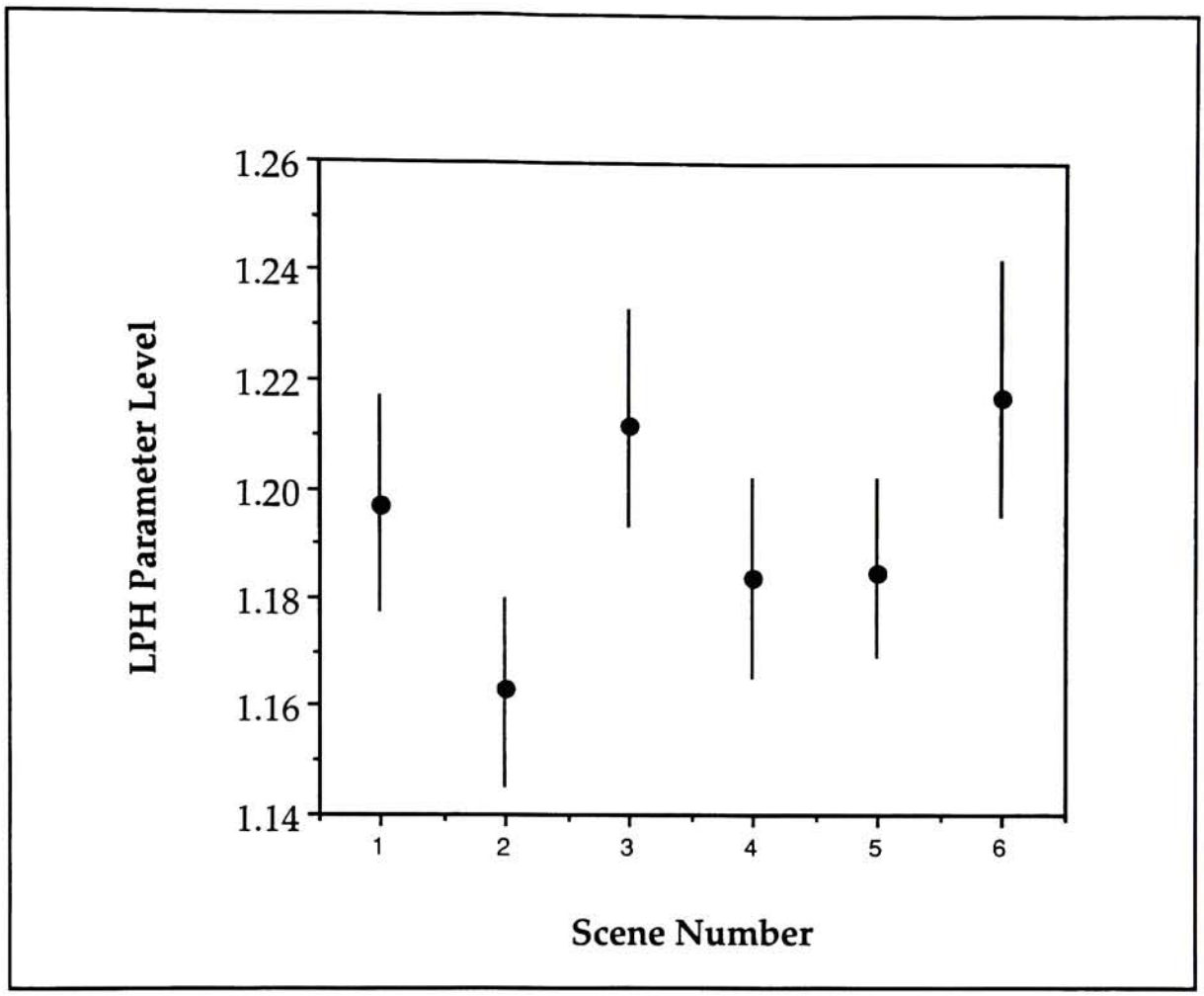
Table 5.5-1 : Acceptability Results by Scene

| Scene | # | Function | T50 | LOWER | UPPER | Prob > Chi $^2$ |
|---|---|---|---|---|---|---|
| gcmpnl | 1 | LMF | 0.91 | 0.90 | 0.92 | 0.12 |
| mdfnfh | 2 | LMF | 0.90 | 0.89 | 0.91 | 0.48 |
| q60mfh | 3 | LMF | 0.89 | 0.88 | 0.90 | 0.22 |
| q60pfh | 4 | LMF | 0.91 | 0.90 | 0.91 | 0.68 |
| rcimnl | 5 | LMF | 0.94 | 0.92 | 0.96 | 0.02 |
| smannl | 6 | LMF | 0.89 | 0.88 | 0.89 | 0.74 |
| gcmpnl | 1 | LPH | 1.20 | 1.18 | 1.22 | 0.69 |
| mdfnfh | 2 | LPH | 1.16 | 1.14 | 1.18 | 0.25 |
| q60mfh | 3 | LPH | 1.21 | 1.19 | 1.23 | 0.23 |
| q60pfh | 4 | LPH | 1.18 | 1.16 | 1.20 | 0.78 |
| rcimnl | 5 | LPH | 1.18 | 1.17 | 1.20 | 0.91 |
| smannl | 6 | LPH | 1.22 | 1.19 | 1.24 | 0.20 |
| gcmpnl | 1 | LPL | 0.86 | 0.85 | 0.87 | 0.83 |
| mdfnfh | 2 | LPL | 0.88 | 0.87 | 0.90 | 0.19 |
| q60mfh | 3 | LPL | 0.86 | 0.84 | 0.87 | 0.14 |
| q60pfh | 4 | LPL | 0.87 | 0.86 | 0.88 | 0.41 |
| rcimnl | 5 | LPL | 0.86 | 0.85 | 0.87 | 0.11 |
| smannl | 6 | LPL | 0.86 | 0.85 | 0.88 | 0.76 |
| gcmpnl | 1 | LSH | 1.30 | 1.28 | 1.34 | 0.19 |
| mdfnfh | 2 | LSH | 1.38 | 1.34 | 1.42 | 0.50 |
| q60mfh | 3 | LSH | 1.40 | 1.37 | 1.44 | 0.17 |
| q60pfh | 4 | LSH | 1.30 | 1.27 | 1.32 | 0.64 |
| rcimnl | 5 | LSH | 1.38 | 1.34 | 1.42 | 0.33 |
| smannl | 6 | LSH | 1.34 | 1.31 | 1.38 | 0.29 |
| gcmpnl | 1 | LSL | 0.84 | 0.83 | 0.86 | 0.31 |
| mdfnfh | 2 | LSL | 0.83 | 0.81 | 0.84 | 0.52 |
| q60mfh | 3 | LSL | 0.82 | 0.80 | 0.83 | 0.19 |
| q60pfh | 4 | LSL | 0.86 | 0.84 | 0.87 | 0.33 |
| rcimnl | 5 | LSL | 0.82 | 0.81 | 0.84 | 0.95 |
| smannl | 6 | LSL | 0.84 | 0.83 | 0.86 | 0.67 |
| gcmpnl | 1 | CMF | 0.83 | 0.81 | 0.84 | 0.60 |

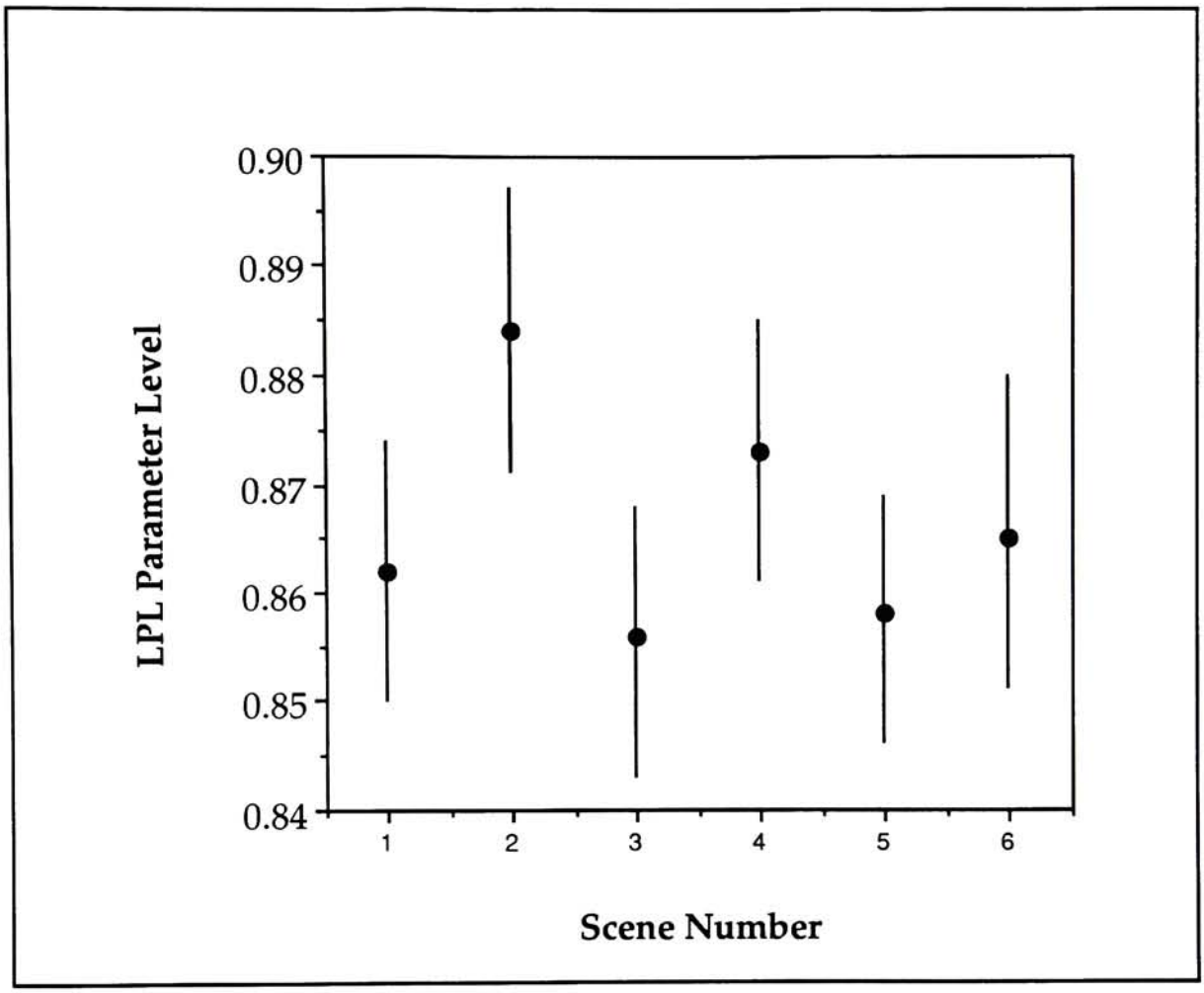
| | | | | | | |
|---|---|---|---|---|---|---|
| mdfnfh | 2 | CMF | 0.85 | 0.83 | 0.86 | 0.24 |
| q60mfh | 3 | CMF | 0.85 | 0.83 | 0.87 | 0.43 |
| q60pfh | 4 | CMF | 0.83 | 0.81 | 0.85 | 0.68 |
| rcimnl | 5 | CMF | 0.91 | . | . | 0.00 |
| smannl | 6 | CMF | 0.83 | 0.81 | 0.85 | 0.18 |
| gcmpnl | 1 | CPH | 1.20 | 1.18 | 1.22 | 0.25 |
| mdfnfh | 2 | CPH | 1.19 | 1.18 | 1.21 | 0.18 |
| q60mfh | 3 | CPH | 1.19 | 1.18 | 1.21 | 0.71 |
| q60pfh | 4 | CPH | 1.17 | 1.15 | 1.19 | 0.51 |
| rcimnl | 5 | CPH | 1.11 | 1.10 | 1.13 | 0.25 |
| smannl | 6 | CPH | 1.20 | 1.18 | 1.22 | 0.30 |
| gcmpnl | 1 | CPL | 0.86 | 0.85 | 0.87 | 0.99 |
| mdfnfh | 2 | CPL | 0.77 | 0.75 | 0.79 | 0.44 |
| q60mfh | 3 | CPL | 0.75 | 0.72 | 0.77 | 0.69 |
| q60pfh | 4 | CPL | 0.79 | 0.76 | 0.81 | 0.52 |
| rcimnl | 5 | CPL | 0.84 | 0.77 | 0.89 | 0.00 |
| smannl | 6 | CPL | 0.80 | 0.78 | 0.82 | 0.13 |
| gcmpnl | 1 | HOH | 10.17 | 9.32 | 11.12 | 0.64 |
| mdfnfh | 2 | HOH | 7.80 | 7.02 | 8.60 | 0.46 |
| q60mfh | 3 | HOH | 9.99 | 9.12 | 10.97 | 0.57 |
| q60pfh | 4 | HOH | 10.66 | 9.83 | 11.60 | 0.73 |
| rcimnl | 5 | HOH | 5.25 | -0.36 | 8.64 | 0.00 |
| smannl | 6 | HOH | 11.45 | 10.48 | 12.65 | 0.35 |
| gcmpnl | 1 | HOL | -8.21 | -15.33 | 3.68 | 0.00 |
| mdfnfh | 2 | HOL | -6.36 | -7.99 | -4.28 | 0.03 |
| q60mfh | 3 | HOL | -9.83 | -12.48 | -7.08 | 0.02 |
| q60pfh | 4 | HOL | -9.65 | -11.85 | -7.37 | 0.03 |
| rcimnl | 5 | HOL | -5.37 | . | . | 0.00 |
| smannl | 6 | HOL | -14.90 | -16.50 | -13.64 | 0.37 |



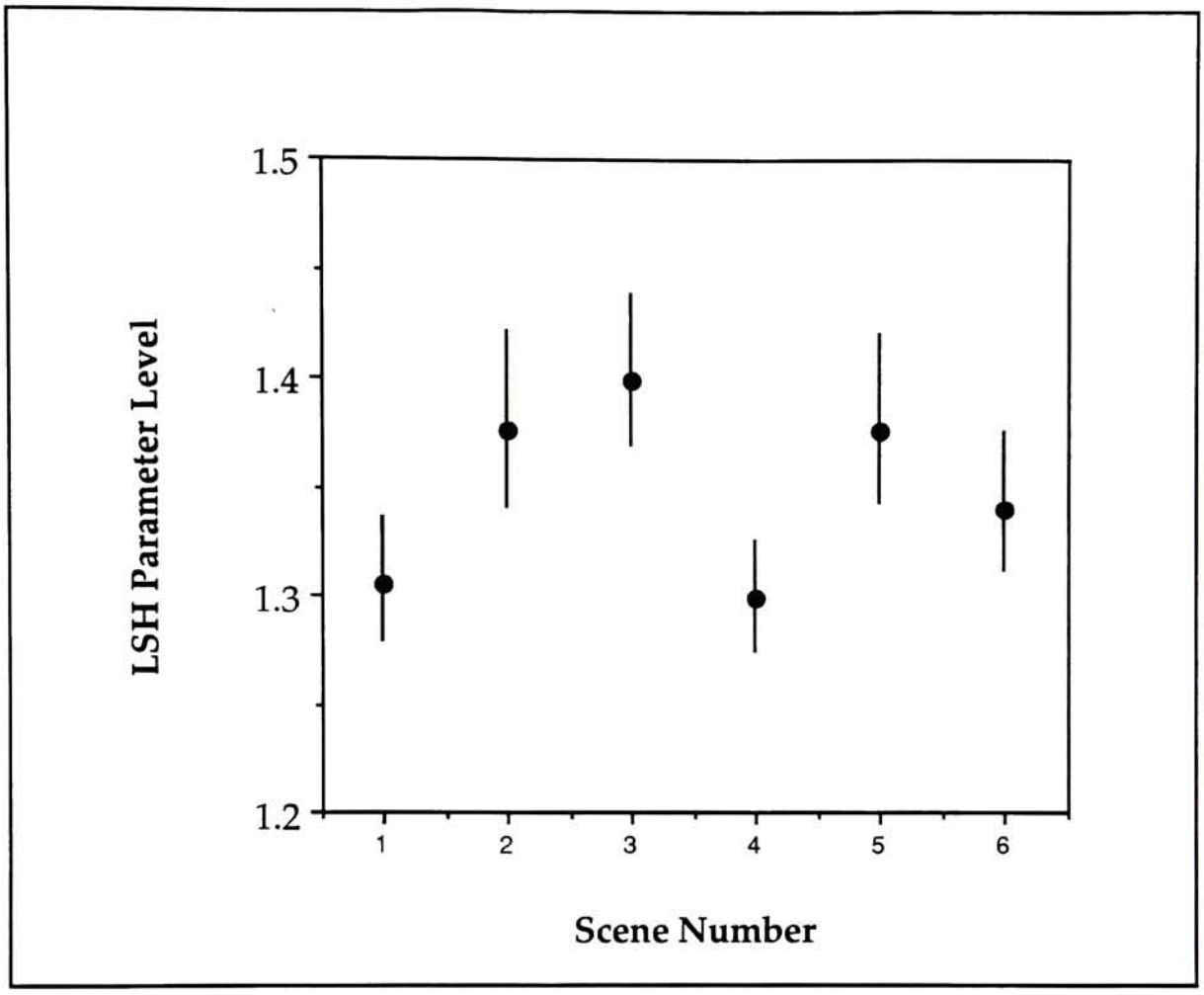

Graph 5.5-1 : Acceptability by Scene for Lightness Multiplicative Factor



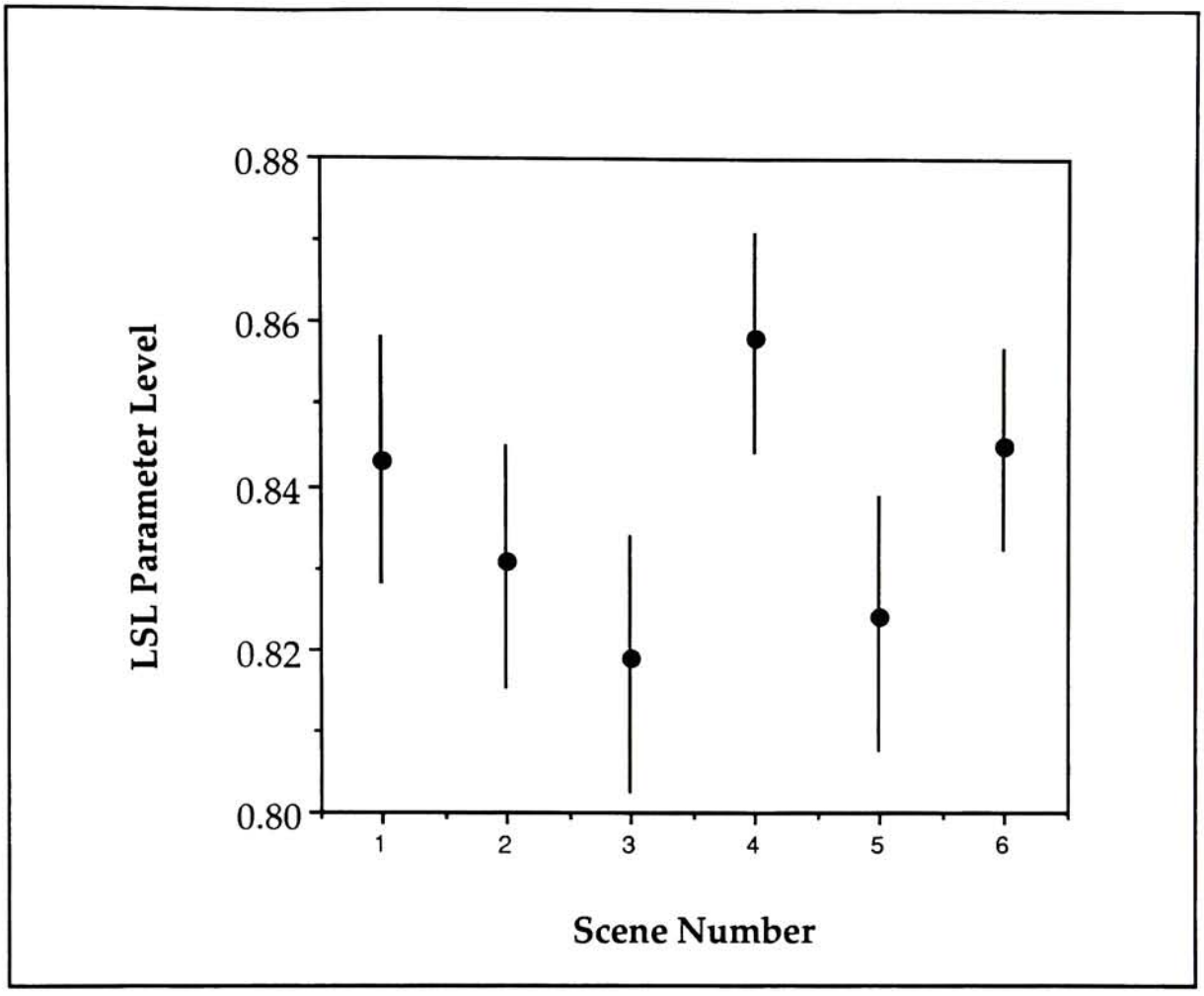

Graph 5.5-2 : Acceptability by Scene for Lightness High Power



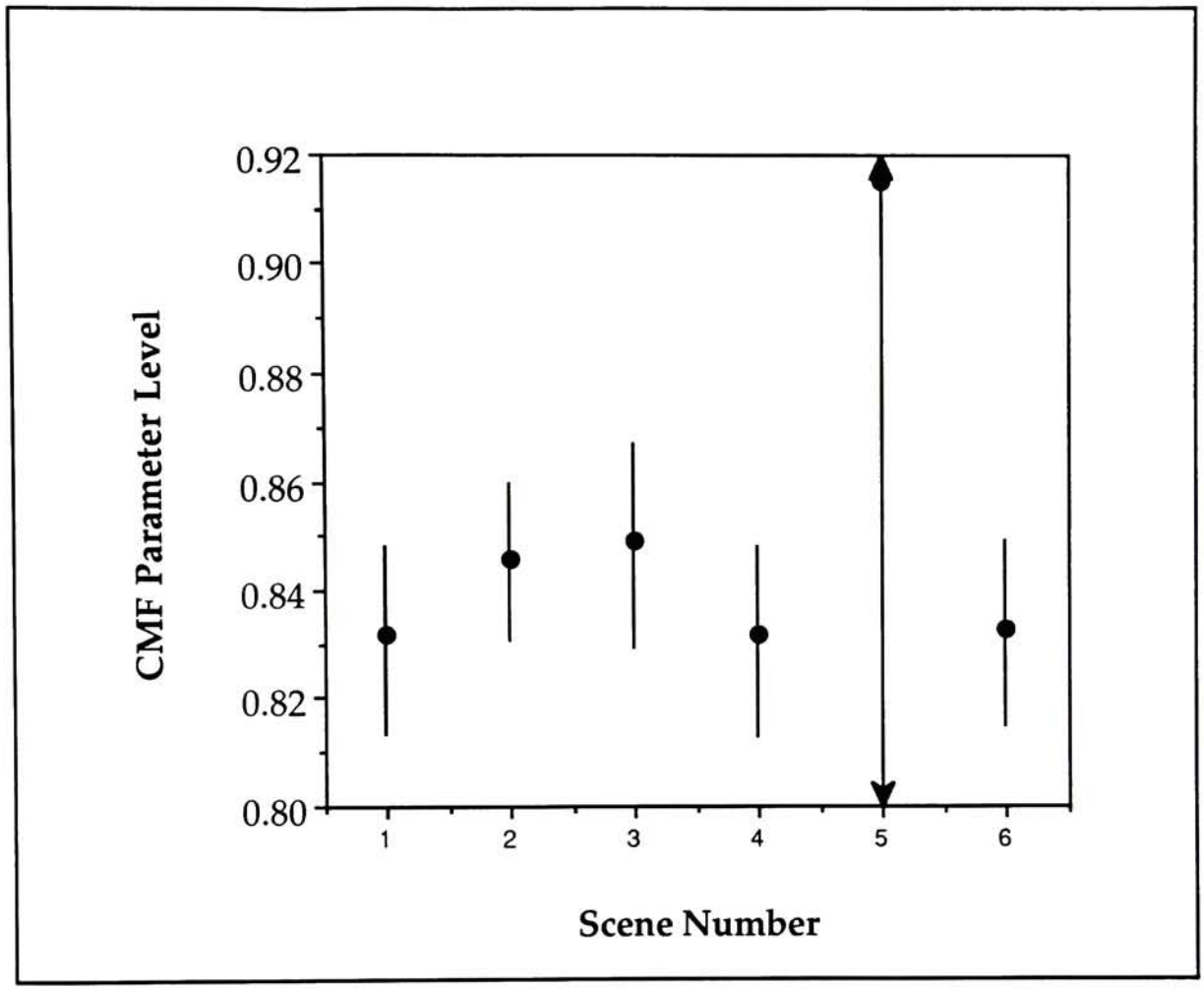

Graph 5.5-3 : Acceptability by Scene for Lightness Low Power



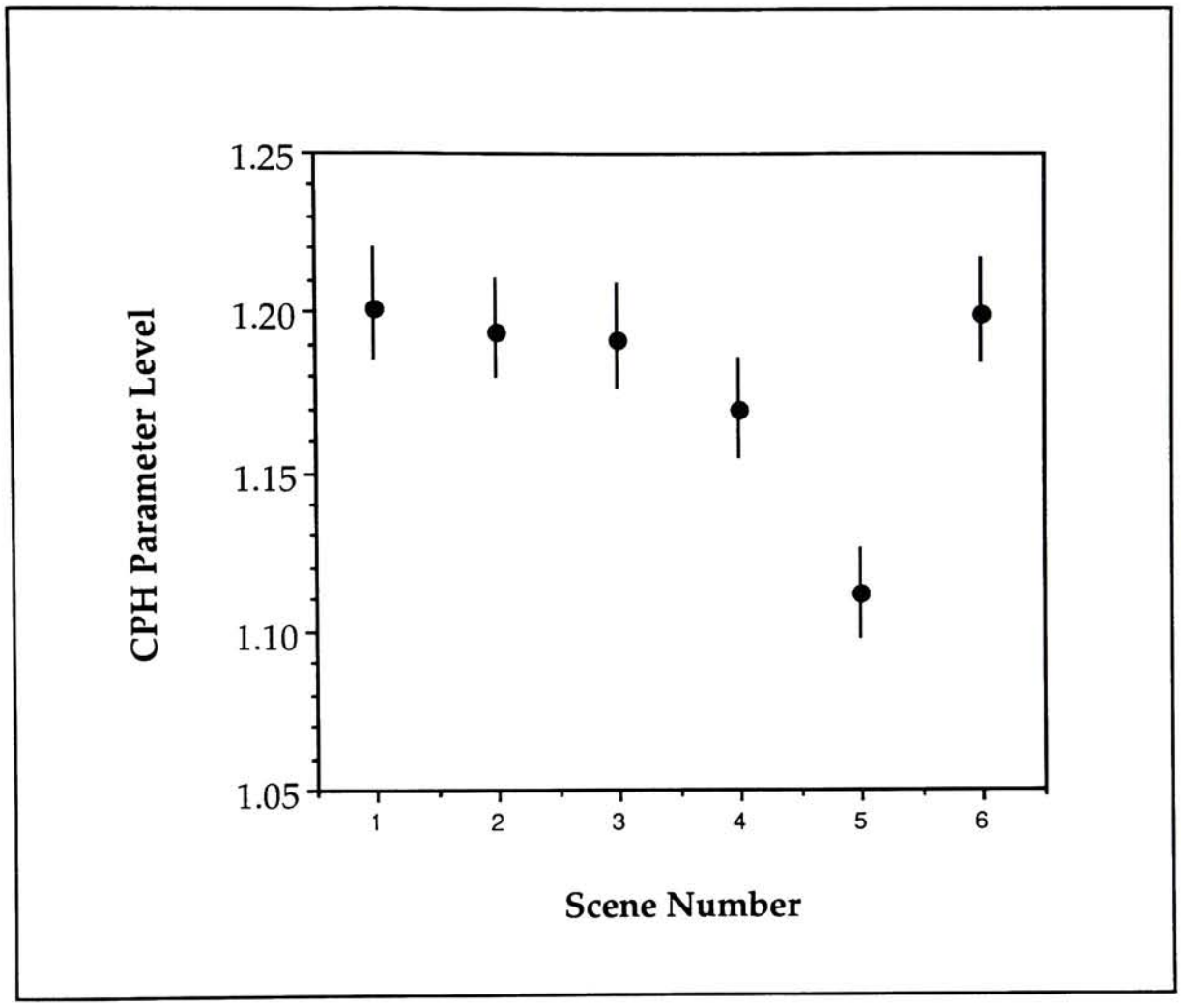

Graph 5.5-4 : Acceptability by Scene for Lightness High Sigmoidal



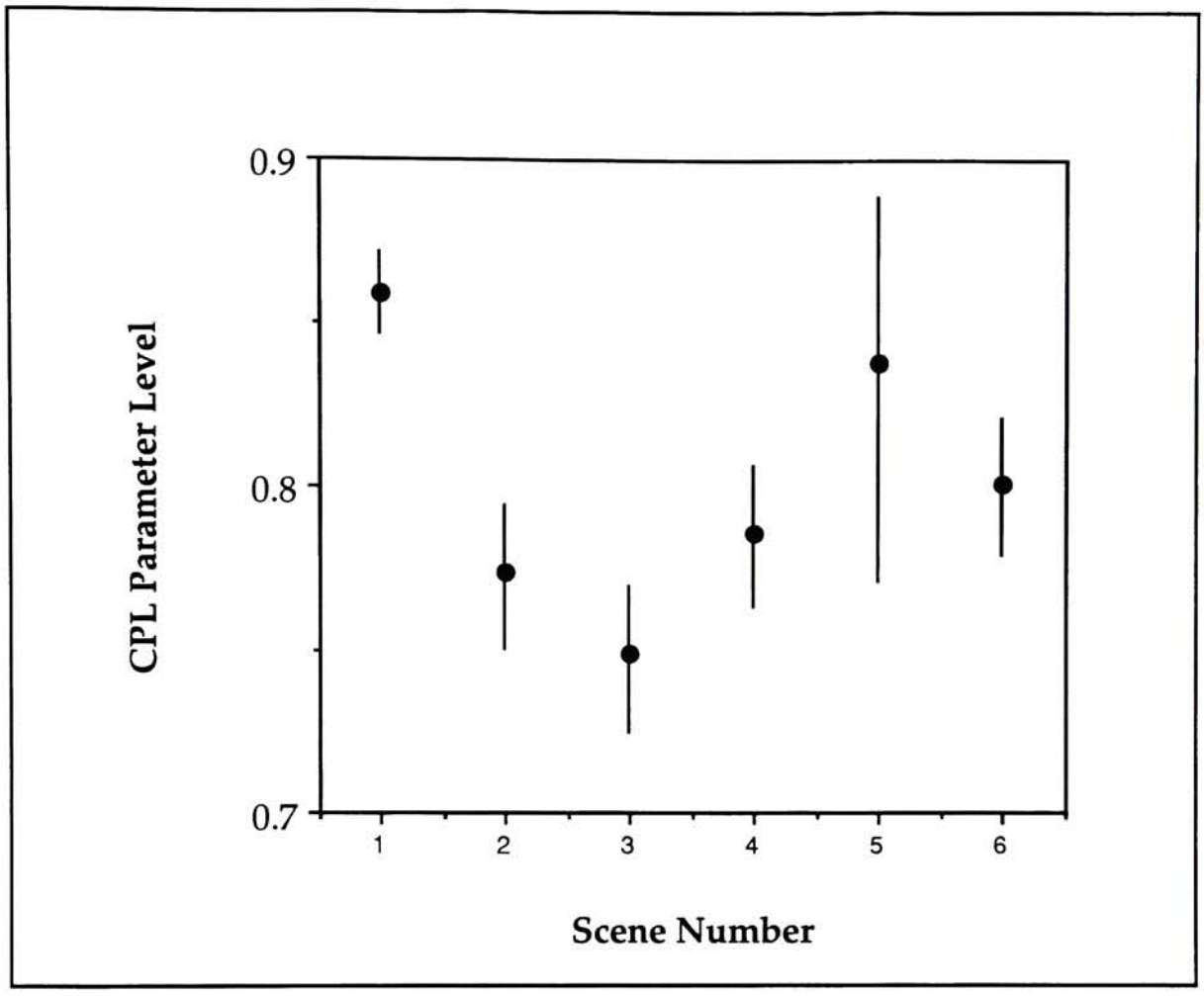

Graph 5.5-5 : Acceptability by Scene for Lightness Low Sigmoidal



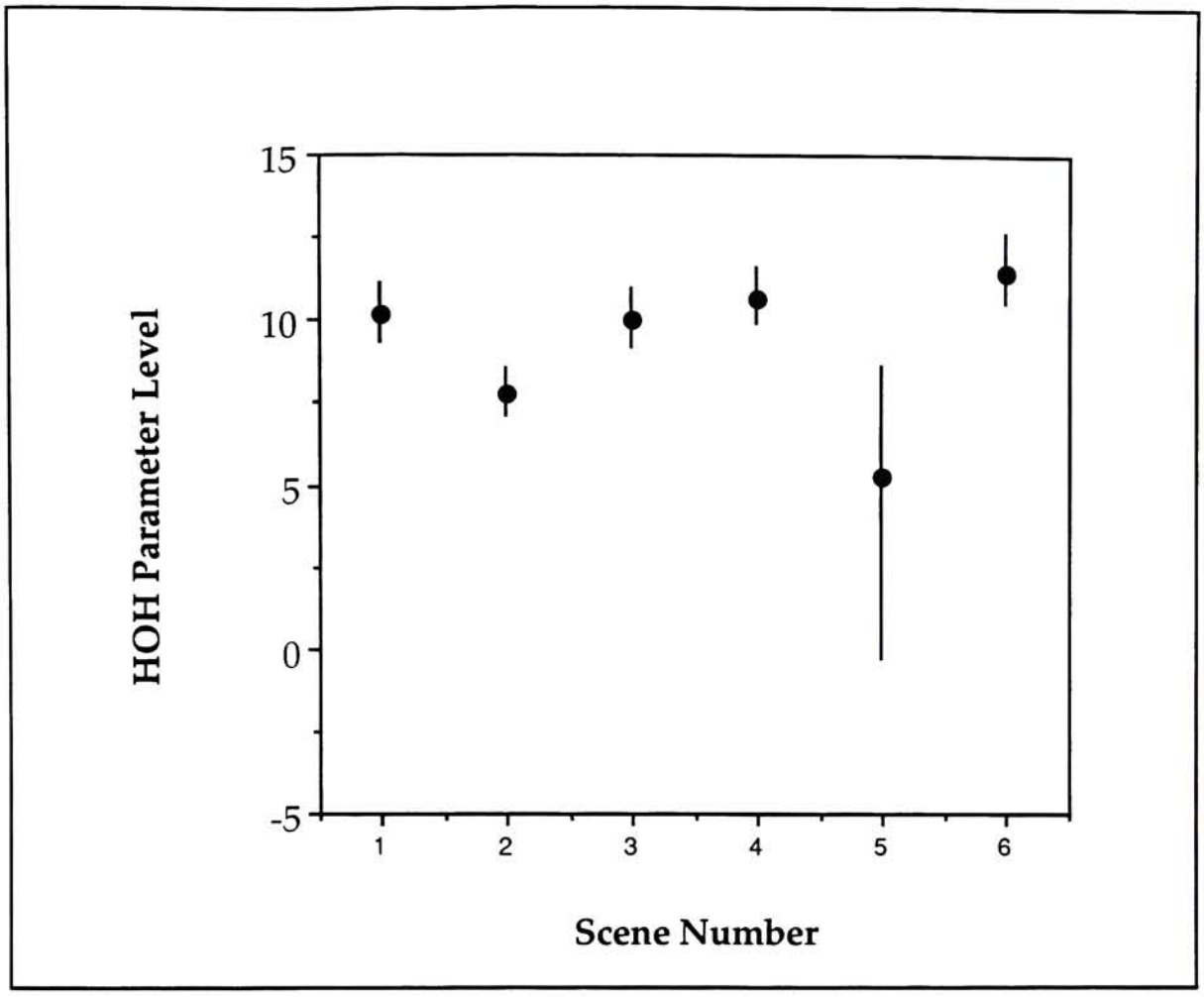

Graph 5.5-6 : Acceptability by Scene for Chroma Multiplicative Factor



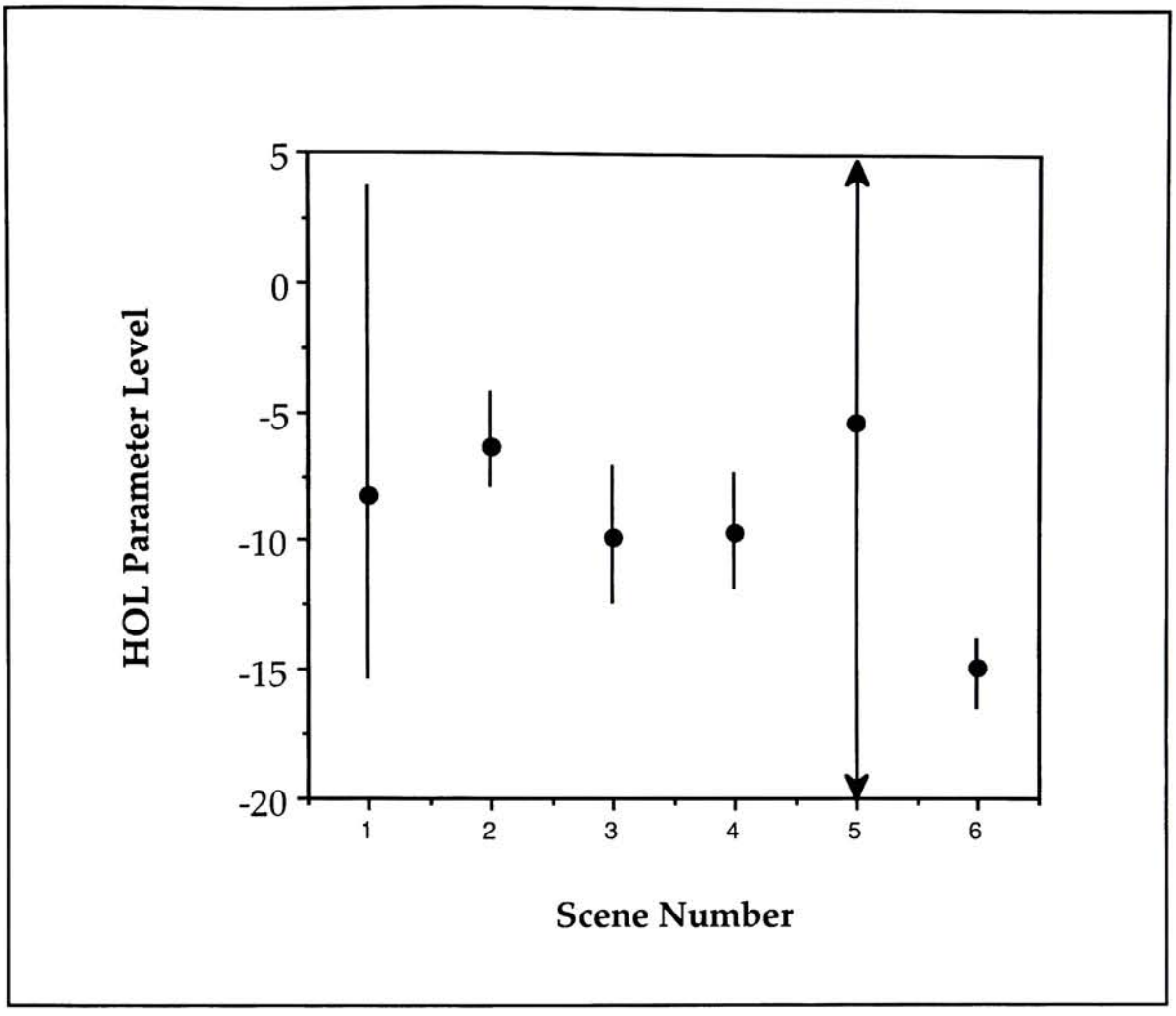

Graph 5.5-7 : Acceptability by Scene for Chroma High Power




Graph 5.5-8 : Acceptability by Scene for Chroma Low Power




Graph 5.5-9 : Acceptability by Scene for Hue Angle Positive Offset




Graph 5.5-10 : Acceptability by Scene for Hue Angle Negative Offset

## 5.6 Comparisons with Color Difference Formulas

The CIELAB, CMC and MCSL color difference equations were evaluated in comparison with the perceptibility tolerances above. Each image was manipulated by the ten transfer functions with parameter levels equal to the T50, UPPER, and LOWER perceptibility values for each scene. The average value of each color dimension was then calculated and from these averages, a color difference was computed. By normalizing the color differences for each manipulated image above, several color difference

formulas can be compared against each other. Tables 5.6-1 and 5.6-2 and graphs 5.6-1 through 5.6-3 show the CIELAB, CMC, and MCSL color difference formulas with respect to the various transfer function tolerances. Ideally, all of the normalized results would overlap at or near unity. The tables and graphs clearly show that this is not the case for the CMC and MCSL color difference formulas and thus these color difference formulas do not adequately predict image tolerances. This is most likely due to the optimization of the CMC and MCSL formulas using small color differences. This optimization reduces the tone reproduction information and thus separates the hue and chroma information from the lightness information. The CIELAB formula performed does adequately predict image tolerances. Since CIELAB is based on the Munsell Color System which uses large color differences, this result indicates that observers judge images using large color differences.

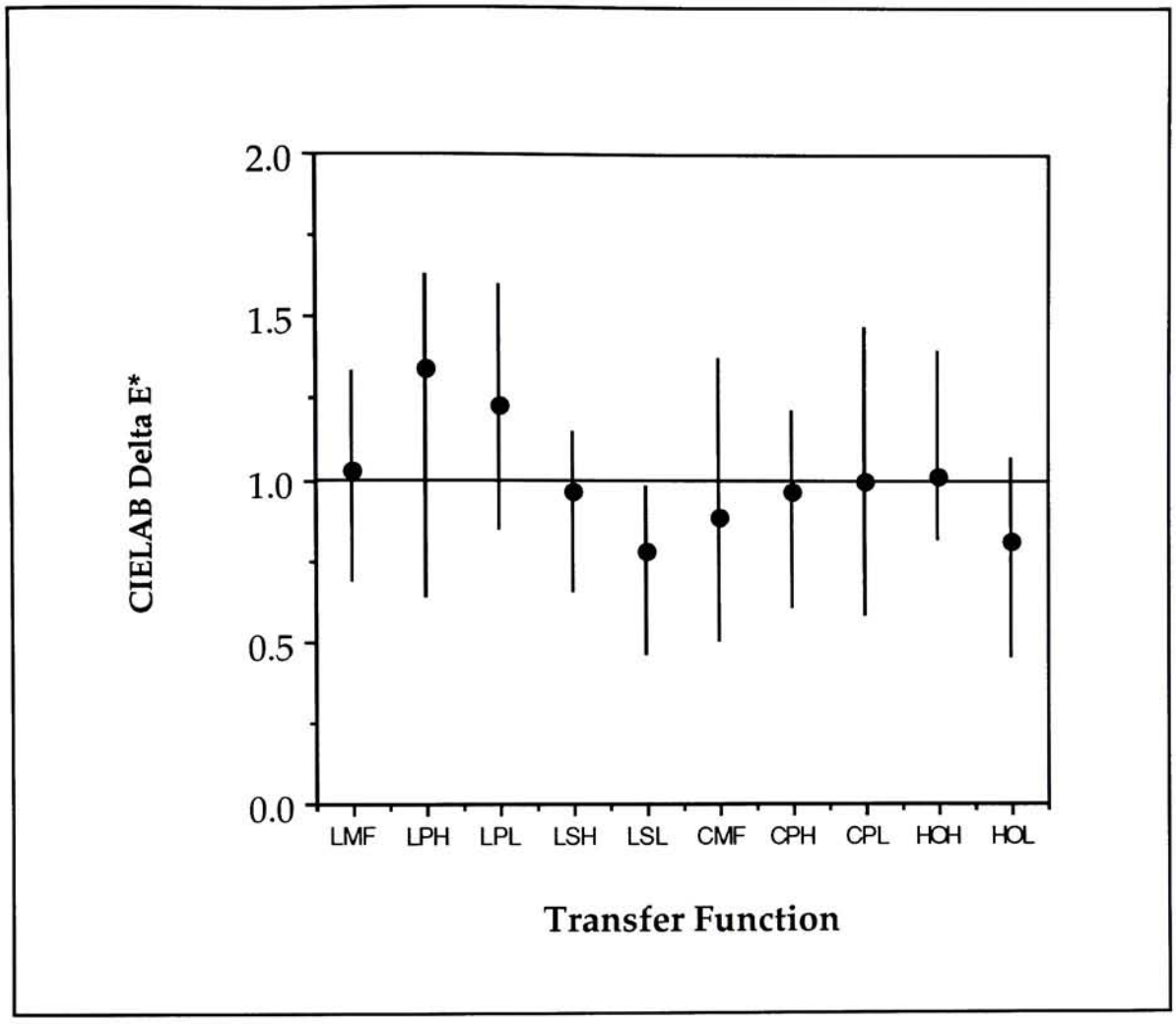
A color tolerance could be derived using a CIELAB color difference of 2.01 (with an uncertainty from 1.11 to 4.06) to give a rough indication of perceptible differences. A value of 6.6 would likewise give a rough indication of acceptable differences. This acceptability result agrees well with previous results derived in a dramatically difference fashion by Stamm.[102]

Table 5.6-1 : Comparison of Raw Color Difference Formulas

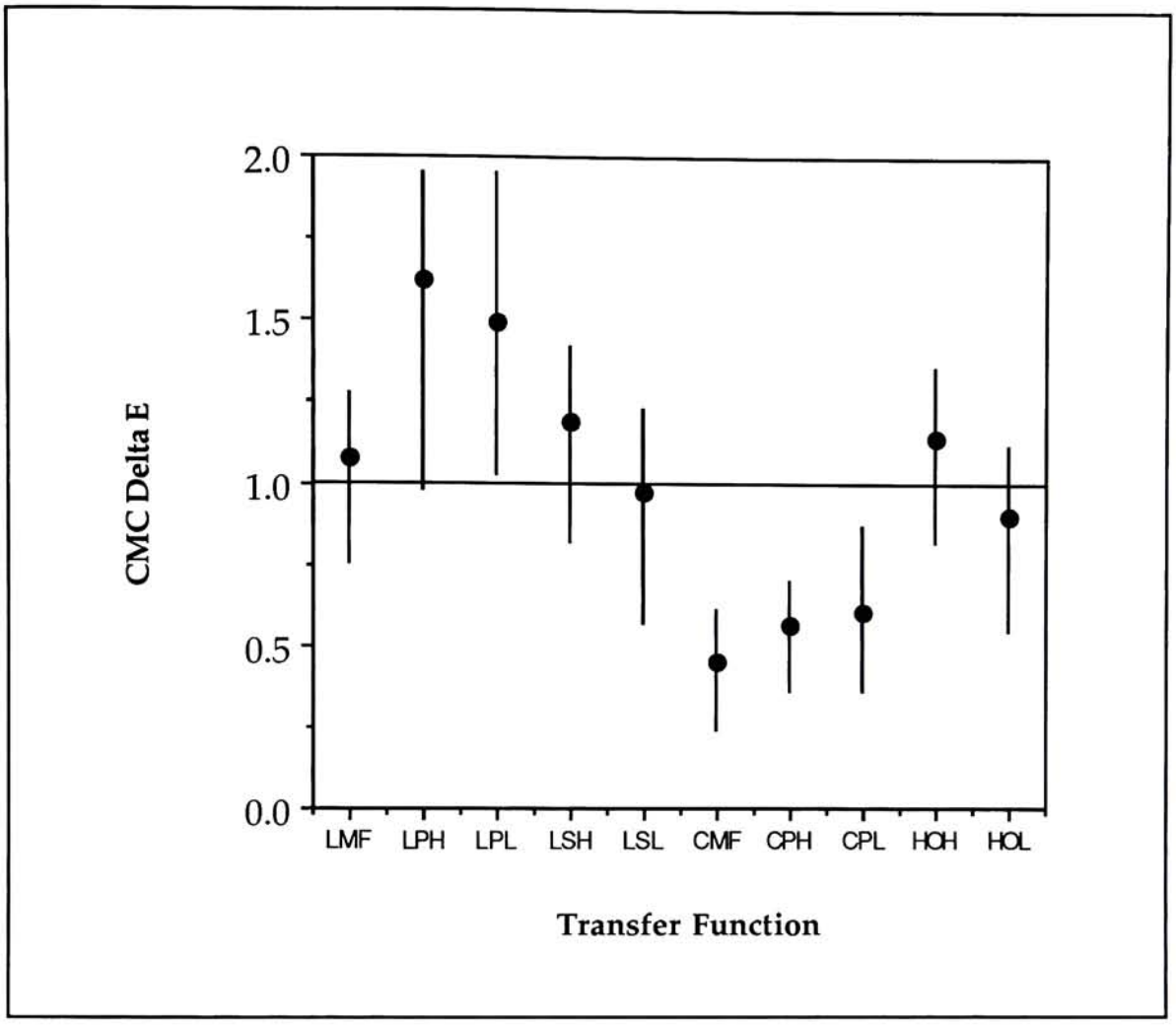
| | CIELAB | | | CMC | | | MCSL | | |
|---|---|---|---|---|---|---|---|---|---|
| Function | T50 | lower | upper | T50 | lower | upper | T50 | lower | upper |
| LMF | 2.56 | 1.70 | 3.32 | 2.74 | 1.90 | 3.26 | 2.56 | 1.72 | 3.07 |
| LPH | 3.32 | 1.57 | 4.06 | 4.11 | 2.45 | 4.94 | 3.32 | 1.94 | 4.02 |
| LPL | 3.07 | 2.10 | 3.99 | 3.80 | 2.59 | 4.95 | 3.07 | 2.10 | 3.99 |
| LSH | 2.38 | 1.61 | 2.85 | 3.02 | 2.03 | 3.62 | 2.38 | 1.61 | 2.85 |
| LSL | 1.94 | 1.14 | 2.44 | 2.47 | 1.41 | 3.13 | 1.94 | 1.14 | 2.44 |
| CMF | 2.19 | 1.23 | 3.41 | 1.16 | 0.58 | 1.54 | 0.94 | 0.48 | 1.26 |
| CPH | 2.39 | 1.50 | 3.00 | 1.41 | 0.88 | 1.77 | 1.15 | 0.72 | 1.45 |
| CPL | 2.49 | 1.45 | 3.64 | 1.53 | 0.90 | 2.21 | 1.24 | 0.73 | 1.80 |
| HOH | 2.52 | 2.01 | 3.47 | 2.89 | 2.06 | 3.45 | 1.80 | 1.28 | 2.16 |
| HOL | 2.01 | 1.11 | 2.66 | 2.30 | 1.37 | 2.84 | 1.43 | 0.85 | 1.77 |
| | | | | | | | | | |
| Average | 2.49 | 1.54 | 3.29 | 2.54 | 1.62 | 3.17 | 1.98 | 1.26 | 2.48 |
| Std. dev. | 0.43 | 0.34 | 0.54 | 0.98 | 0.69 | 1.17 | 0.82 | 0.57 | 0.99 |

Table 5.6-2 : Comparison of Normalized Color Difference Formulas

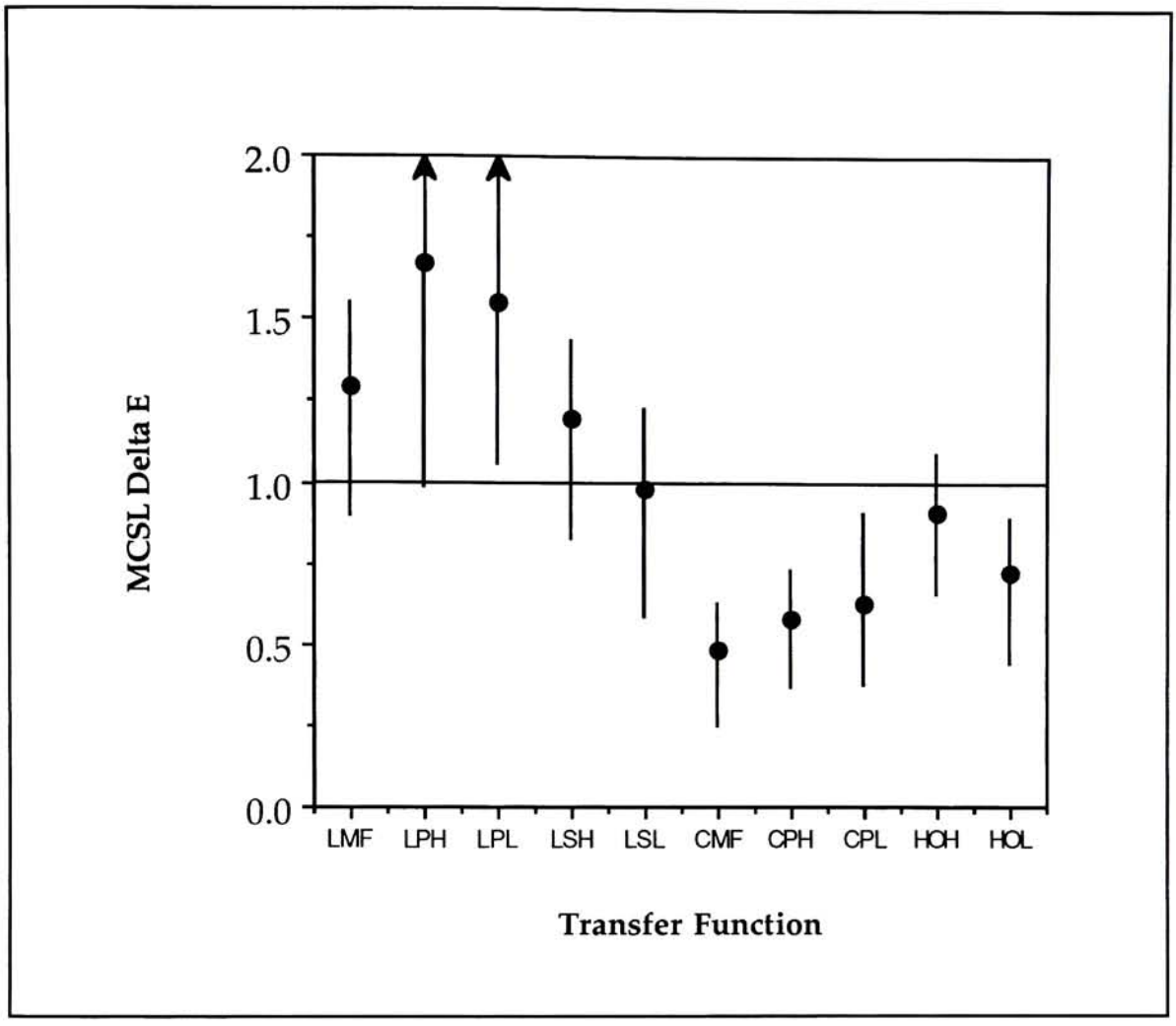
| | CIELAB | | | CMC | | | MCSL | | |
|---|---|---|---|---|---|---|---|---|---|
| Function | T50 | lower | upper | T50 | lower | upper | T50 | lower | upper |
| LMF | 1.03 | 0.68 | 1.33 | 1.08 | 0.75 | 1.28 | 1.29 | 0.87 | 1.55 |
| LPH | 1.34 | 0.63 | 1.63 | 1.62 | 0.97 | 1.95 | 1.67 | 0.98 | 2.02 |
| LPL | 1.23 | 0.84 | 1.60 | 1.49 | 1.02 | 1.95 | 1.55 | 1.06 | 2.01 |
| LSH | 0.96 | 0.65 | 1.15 | 1.19 | 0.80 | 1.42 | 1.20 | 0.81 | 1.44 |
| LSL | 0.78 | 0.46 | 0.98 | 0.97 | 0.56 | 1.23 | 0.98 | 0.57 | 1.23 |
| CMF | 0.88 | 0.49 | 1.37 | 0.45 | 0.23 | 0.61 | 0.48 | 0.24 | 0.63 |
| CPH | 0.96 | 0.60 | 1.21 | 0.56 | 0.35 | 0.70 | 0.58 | 0.36 | 0.73 |
| CPL | 1.00 | 0.58 | 1.47 | 0.60 | 0.35 | 0.87 | 0.63 | 0.37 | 0.91 |
| HOH | 1.01 | 0.81 | 1.40 | 1.14 | 0.81 | 1.36 | 0.91 | 0.65 | 1.09 |
| HOL | 0.81 | 0.45 | 1.07 | 0.90 | 0.54 | 1.12 | 0.72 | 0.43 | 0.89 |
| | | | | | | | | | |
| Average | 1.00 | 0.62 | 1.32 | 1.00 | 0.64 | 1.25 | 1.00 | 0.63 | 1.25 |
| Std. dev. | 0.17 | 0.14 | 0.22 | 0.39 | 0.27 | 0.46 | 0.42 | 0.28 | 0.50 |




Graph 5.6-1: Normalized CIELAB Color Difference Results




Graph 5.6-2: Normalized CMC Color Difference Results




Graph 5.6-3: Normalized MCSL Color Difference Results

## 5.7 Application of the Tolerances

The actual tolerances can be used to evaluate other image manipulation routines such as gamut compression or quantization. The are two possible methods of applying the experimental tolerances in order to derive perceptibility and acceptability data without visually assessing the images. The first method incorporates non-linear regression and is described in detail in Appendix K. The second method uses the CIELAB color difference results with three possible outcomes, below tolerance, above

tolerance, and inconclusive.

A comparison can be performed between an original and a manipulated image using nonlinear regression with the various transfer functions operating on the manipulated image. The resulting regression parameter estimates can be compared directly to their respective tolerances. This comparison establishes whether the manipulated image is perceptibly or acceptably different from the original. An SAS code example of this method is shown in Appendix K. In this example an image was manipulated in RGB device space using a simple power function on all three dimensions with a value of 1.2. The manipulated image was analyzed to be both perceptibly and unacceptably different in lightness from the original. In addition the changes in chroma were imperceptible.

A second comparison method uses the CIELAB color difference results. By using the minimum and maximum differences, limits can be established for classifying image differences. The CIELAB minimum and maximum differences are 1.11 and 4.06 respectively. Actual differences are calculated by averaging the color differences for each pixel in an image. If the actual difference is below the minimum difference, then the manipulated image is not perceptibly different from the original. If the color difference is above the maximum difference, then the manipulated image is perceptibly different from the original. If neither of the above two cases are true, then this method does not yield conclusive results.

# 6.0 Conclusions and Summary

In conclusion, this thesis has established a solid and fertile research environment, established perceptibility and acceptability tolerances for several images, determined that scene content does not significantly affect these tolerances, determined that the CIELAB color difference equation does adequately model color differences in images, while the CMC and MCSL color differences equations do not and developed a method to apply these tolerances for future research.

The hardware environment could be improved with several modifications and additions. The signal to noise ratio of the Howtek scanner could be improved by installing a linear power supply. The scanner currently uses a switching power supply that artificially decreases the signal to noise ratio by approximately 2 bits of the 8 bit signal.[103] Spatial non-uniformity is also a significant problem in the present system. A spatial compensation routine could help resolve the loss of signal toward each side of the scanned image at the cost of signal resolution. Additionally, using originals no larger than 6 inches would avoid the worst of the signal loss. While substantially improving the current performance of the scanner, these modifications would not create the high quality scanned image necessary for future research. Instead a desktop drum scanner will be necessary. In addition to significantly improved signal and uniformity, a drum scanner could be modified to be colorimetric. A high quality printer and film recorder are needed to perform hardcopy/softcopy experiments and produce accurate presentation results. Finally, improvements to the HP workstation are necessary to improve computational speeds, including increased RAM memory and a CPU upgrade to a 68040. If these suggestions are implemented,

in addition to the software changes outlined below, the research environment would be greatly enhanced.

The software environment needs several additions before more extensive research can be performed. Various optimizations can be performed to reduce these hardware requirements, including replacing the power and trigonometric functions by lookup tables, and using non-portable device-dependent optimizations. Alternately, a fixed-precision format could be employed with additional loss in precision. More research needs to be performed to determine the necessary precision needed to avoid perceptual loss in computations. Numerous additional routines need to be implemented. Cubic and tetrahedral interpolation functions are needed to compare analytical models to empirical models. Routines to implement various chromatic adaptation models, including those of Von Kries[104], Hunt,[10], and Nayatani[11], are needed to investigate the effects of chromatic adaptation. Various cone response spaces, along with the Yu'v' chromaticity space, Munsell space and the 1990 OSA space are needed to evaluate the uniformity of color spaces. Extensive work on gamut mapping techniques is necessary. As more applications are explored, the appropriate routines can be implemented.

The lack of scene dependency was surprising and contradicted previous results. Yet none of the previous results were based on pictorial images and changes intuitive perceptual dimensions. This result is even more surprising since the six scenes were specifically chosen to appear significantly different.

One possible weakness in the experimental design is the small number of scenes used. This can be rectified by including different scenes in future experiments to verify the current results. With improvements in hardware and elimination of functions using symmetry, eight images using three

functions is now estimated to take well under one hour of observation time.

Possible future applications include: comparing tolerances of color checker charts with those of pictorial images, using dual monitors to measure image separation effects on colorimetric tolerances, using a digital printer to establish tolerances for hardcopy/softcopy reproductions, comparing different gamut mapping techniques, comparing different chromatic adaptation models and comparing different experimental techniques.

Several recent articles provide color reproduction factors based on color checker charts. This approach can be validated by measuring the visual tolerances of various color checker charts and mondrians, and comparing the results with those using pictorial images.

Over thirty years ago, Neugebauer described the need to measure the effects of image separation on color reproduction tolerances. This environment, using dual display monitors, could effectively measure the colorimetric effects of image separation. By measuring these effects in a purely softcopy environment, they could be accounted for in hardcopy/softcopy experiments where they are unavoidable.

The TC1-27 CIE committee is establishing guidelines on experiments in hardcopy/softcopy color reproduction. This environment should fit well within these guidelines and be able to provide immediate and ongoing results for use in evaluating various issues in hardcopy/softcopy color reproduction. The issues include comparing different proofing materials such as chromalin and dye transfer materials and comparing different white points.

Color electronic prepress systems have been gaining popularity in recent years. This environment could be used to determine the optimum setup of these systems with respect to such issues as surround, image

separation, and illumination.

Over the last few years, the CIE has been reviewing chromatic adaptation and color appearance models. This environment could provide experimental evidence comparing the models of Hunt, Nayatani, Fairchild and others.

While these are just a few of the possibilities, others include comparing different gamut mapping techniques, evaluating the effects of different types of image noise, and comparing different compression algorithms, such as lightness weighted and inversely exponential chromatic compression. As this environment matures, we expect even more possibilities to arise.

Research in this area is important because it has been virtually overlooked or held proprietary until now and is of paramount importance in trying to establish color reproduction with electronic imaging. Most importantly, a step toward adhering to the advice of MacAdam and Neugebauer has been made after decades of silence and the fundamental problems of electronic color image reproduction cannot be solved without adhering to their advice.

## References :


[1] D. L. MacAdam, Visual sensitivities to color differences in daylight, *J. Opt. Soc. Am.* **32**, 247-274 (1942).

[2] S. M. Newall, Preliminary Report of the O.S.A Subcommittee on the Spacing of the Munsell Colors, *J. Opt. Soc. Am*, **30**, 617-645 (1940).

[3] S. M. Newall, D. Nickerson, and D. B. Judd, Final Report of the O.S.A Subcommittee on the Spacing of the Munsell Colors, *J. Opt. Soc. Am*, **33**, 385-418 (1943).

[4] CIE, *Colorimetry, Second Edition* (Official Recommendations of the International Commission on Illumination), CIE Publ. 15.2, Central Bureau of the CIE, Vienna (1986).

[5] R. McDonald, Industrial Pass/Fail Color Matching. Part III - Development of Pass/Fail Formula for Use with Instrumental Measurement of Colour Difference, *J. Soc. Dyers Col.* **96**, 486-495 (1980).

[6] R. McDonald, Acceptability and perceptibility decisions using the CMC color difference formula, *Tex. Chem. Col.* **20** No. 6, 31-37 (1988).

[7] R. McDonald, European Practices and Philosophy in Industrial Colour-difference Evaluation, *Color Res. Appl.* **15**, 249-260 (1990).

[8] R. G. Kuehni, Industrial Color Difference: Progress and Problems, *Color Res. Appl.*, **15**, 261-265 (1990).

[9] R. S. Berns, D. H. Alman, and L. Reniff, Visual Determination of Supra-Threshold Color-Difference Tolerances Using Probit Analysis, Paper presented at the ISCC Conference on Color Discrimination Psychophysics, Williamsburg, Virginia, November 28-31, (1989).

[10] R. W. G. Hunt, A Model of Colour Vision for Predicting Colour Appearance in Various Viewing Conditions, *Color Res. Appl.*, **12**, 297-314 (1987).

[11] Y. Nayatani, K. Takahama, and H. Sobagaki, Prediction of Color Appearances Under Various Adapting Conditions, *Color Res. Appl.*, **11**, 62-71 (1986).

[12] M. D. Fairchild, Formulation and Testing of an Incomplete-Chromatic-Adaptation Model,*Color Res. Appl.*, **16**, 243-250 (1991).

[13] L. A. Jones, Photographic Reproduction of Tone, *J. Opt. Soc. Am*, **5**, 232-258 (1921).

[14] L. A. Jones and Nelson, Study of Various Sensitometric Criteria of Negative Film Speeds, *J. Opt. Soc. Am.*, **30**, 93-109 (1940).

[15] Y. A. C. Yule, Theory of Subtractive Color Photography, II. Prediction of Errors in Color Rendering Under Given Conditions, *J. Opt. Soc. Am.*, **28**, 481-492 (1938).

[16] W. L. Brewer, W. T. Hanson, and C. A. Horton, Subtractive Color Reproduction. The Approximate Reproduction of Selected Colors, *J. Opt. Soc. Am.*, **39**, 924-927 (1949).

[17] W. T. Hanson, Color Correction with Colored Couplers, *J. Opt. Soc. Am.*, **40**, 166-171 (1950).

[18] W. T. Hanson and C. A. Horton, Subtractive Color Reproduction : Interimage Effects, *J. Opt. Soc. Am.*, **42**, 663-669 (1952).

[19] W. T. Hanson and W. L. Brewer, Subtractive Color Photography: The Role of Masks, *J. Opt. Soc. Am.*, **44**, 129-134 (1954).

[20] C. J. Bartleson, Influence of Observer Adaptation on the Acceptance of Color Prints, *Pho. Sci. Eng.*, **2**, 32-39 (1958).

[21] W. W. Woodbury, Two Psychophysical Methods for Evaluating the Quality of Projected Color Slides, *Pho. Sci. Eng.*, **4**, 69-73 (1960).

[22] W. W. Woodbury and C. J. Bartleson, Psychophysical Methods for Evaluating the Quality of Color Transparencies: I. Comparison of Categorial- and Comparitive-Judgement Data, *Pho. Sci. Eng.*,**6**, 10-14 (1962)

[23] C. J. Bartleson and W. W. Woodbury, Psychophysical Methods for Evaluating the Quality of Color Transparencies: II. Control of Observer Adaptation in Categorical Judgments, *Pho. Sci. Eng.*,**6**, 15-18 (1962).

[24] C. J. Bartleson and W. W. Woodbury, Psychophysical Methods for Evaluating the Quality of Color Transparencies: III. Effect of Number of Categories, Anchors, and Types of Instructions on Quality Ratings, *Pho. Sci. Eng.*,**9**, 323-330 (1965).

[25] L. A. Jones and H. R. Condit, The Brightness Scale of Exterior Scenes and the Computation of Correct Photographic Exposure, *J. Opt. Soc. Am.*, **31**, 651-678 (1941).

[26] J. L. Simonds, A Quantitative Study of the Influence of Tone-Reproduction Factors on Picture Quality, *Pho. Sci. Eng.*, **5**, 270-277 (1961).

[27] P. Kowaliski, Tone Reproduction in Colour Scales, *J. Pho. Sci.*, **11**, 169-176 (1963).

[28] R. W. G. Hunt, Luminance Levels in Colour Transparencies and Reflexion Prints, *J. Pho. Sci.*, **13**, 108-114 (1965).

[29] C. J. Bartleson, Interrelations Among Screen Luminance, Camera Exposure, and Quality of Projected Color Transparencies, *Pho. Sci. Eng.*, **9**, 174-178 (1965).

[30] J. Smits and H. Corluy, Colour Reproduction as a Function of Tone Scale in Negative-Positive and Reversal Systems, *J. Pho. Sci.*, **13**, 185-193 (1965).

[31] C. J. Bartleson and R. F. Witzel, Illumination for Color Transparencies, *Pho. Sci. Eng.*, **11**, 329-335 (1967).

[32] C. J. Bartleson and E. J. Breneman, Brightness Perception in Complex Fields, *J. Opt. Soc. Am.*, **57**, 953-957 (1967).

[33] C. J. Bartleson and E. J. Breneman, Brightness Reproduction in the Photographic Process, *Pho. Sci. Eng.*, **11**, 254-262 (1967).

[34] R. W. G. Hunt, The Tone Reproduction of Colour Photographic Materials, *J. Pho. Sci.*, **17**, 198-204 (1969).

[35] W. Behrendt, Tolerances and Probability of the Optimum in Colour Photography, *J. Pho. Sci.*, **13**, 117-124 (1965).

[36] E. J. Breneman, The Effect of Level of Illuminance and Relative Surround Luminance on the Appearance of Black-and-White Photographs, *Pho. Sci. Eng.*, **6**, 172-179 (1962).

[37] C. J. Bartleson, Visual Comparison of Photographic Transparencies and Photomechanical Reflection Prints, *Pho. Sci. Eng.*, **12**, 36-40 (1968).

[38] J. E. Pinney and L. E. DeMarsh, The Study of Colour Reproduction by Computation and Experiment, *J. Pho. Sci.*, **11** 249-255 (1963).

[39] C. J. Bartleson, Color Perception and Color Television, *J. Soc. Mot. Pic. Tel. Eng.*, **77**, 1-11 (1968).

[40] C. J. Bartleson, Some Observations on the Reproduction of Flesh Colors, *Pho. Sci. Eng.*, **3**, 114-117 (1959).

[41] C. J. Bartleson and C. P. Bray, On the Preferred Reproduction of Flesh, Blue-Sky and Green-Grass Colors, *Pho. Sci. Eng.*, **6**, 19-25 (1962).

[42] R. W. G. Hunt, I. T. Pitt, and L. M. Winter, The Preferred Reproduction of Blue Sky, Green Grass and Caucasian Skin in Colour Photography, *J. Pho. Sci.*, **22**, 144-149 (1974).

[43] H. E. J. Neugebauer, Die theoretischen Grundlagen des Mehrfarbenbuchdrucks, *Zeitschrift fur wissenshaftliche Photographie Photophysik und Photochemie*, **36**, 73-89 (1937).

[44] F. Pollak, Masking for Halftone,*J. Phot. Sci.*, **3**, 180-188 (1955).

[45] F. Pollak, Two Masks for Halftone, *J. Phot. Sci.*, **4**, 65-68 (1956).

[46] F. R. Clapper and J. A. C. Yule, Reproduction of Color with Halftone Images, *Tech. Assoc. Graphic Arts.*, **7**, 1-14 (1955).

[47] F. R. Clapper, An Empirical Determination of Halftone Color-Reproduction Requirements,*Tech. Assoc. Graphic Arts.*, **13**, 31-41 (1961).

[48] D. W. Enright, J. R. Huntsman, J. Rubin, and J. Souter, Toward a Colorimetric Specification for GAA/SWOP Proofing and Production Inks - Phase III, *Tech. Assoc. Graphic Arts.*, **42**, 87-103 (1990)

[49] U. Gast, Electronic Reproduction in the 90's, *Tech. Assoc. Graphic Arts.*, **40**, 13-28 (1988)

[50] S. Viggiano, Modeling the Color of Multi-Colored Halftones, *Tech. Assoc. Graphic Arts.*, **42**, 44-62 (1990).

[51] T. Johnson and R. Luo, Colour Appearance Modelling, *Tech. Assoc. Graphic Arts.*, **42**, 144-165 (1990).

[52] F. J. Bingley, Colorimetry in Color Television, *Proc. I.R.E.*, 838-851 (1953).

[53] F. J. Bingley, Colorimetry in Color Television - Part II, *Proc. I.R.E.*, 48-51 (1954).

[54] F. J. Bingley, Colorimetry in Color Television - Part III, *Proc. I.R.E.*, 51-57 (1954).

[55] NTSC, Panel 12, The NTSC Color Television Standards, *Proc. I.R.E.*, 46-48 (1954).

[56] R. P. Burr, The Use of Electronic Masking in Color Television, *Proc. I.R.E.*, 192-200 (1954).

[57] CIE Publication 13.2 1974 *Method of measuring and specifying colour rendering of light sources* (Paris: CIE).

[58] M. R. Pointer, Measuring Colour Reproduction, *J. Pho. Sci.*, **34**, 81-90 (1986).

[59] K. Staes, Evaluation of Pictorial Color-Reproduction Appearance, *Col. Rep. Appl.*, **4**, 5-13 (1979).

[60] B. Hisdal, Color Samples for Colorimetric Fidelity Testing in Television, *Col. Rep. Appl.* **10**, 230-234 (1985).

[61] E. J. Breneman, A Color Chart for Use in Evaluating Quality of Color Reproduction, *Pho. Sci. Eng.*, **1**, 74-78 (1957).

[62] C. S. McCamy, H. Marcus, J. G. Davidson, A Color-Rendition Chart, *J. Appl. Pho. Eng.*, **2**, 95-99 (1976).

[63] D. L. MacAdam, Photographic Aspects of the Theory of Three-Color Reproduction, *J. Opt. Soc. Am.*, **28**, 399-418 (1938).

[64] D. L. MacAdam, Subtractive Color Mixture and Color Reproduction, *J. Opt. Soc. Am.*, **28**, 466-480 (1938).

[65] D. L. MacAdam, Quality of Color Reproduction, *J. Soc. Mot. Pic. Tel. Eng.*, **56**, 487-512 (1951).

[66] H. E J. Neugebauer, The Colorimetric Effect of the Selection of Printing Inks and Photographic Filters on the Quality of Multicolor Reproductions,*Proc. Tech. Assoc. Graphic Arts*, 15-28 (1956).

[67] L. E. DeMarsh, Color Reproduction in Color Television, Proc. I.S.C.C., Williamsburg, Va., 69-97 (1971).

[68] A. Masia, A Color Separating System Based on Principles of Colorimetry,*Proc. Tech. Assoc. Graphic Arts*, 346-361 (1984).

[69] M. Pearson, Image Reproduction Colorimetry, *Col. Res. Appl.*, **11**, 47-50 (1986).

[70] B. Saunders, Visual Matching of Soft Copy and Hard Copy, *J. Imaging Tech.*, **12**, 35-38 (1986).

[71] R. S. Berns and R. J. Motta, Colorimetric Calibration of Soft-Copy Devices to Aid in Hard-Copy Reproduction, *proceedings SPSE 41st annual conferance*, 266-269 (1988).

[72] American National Standards Institute, "American National Standard for Information Systems - Programming Language C, X3.159-1989".

[73] UNIX is a registered trademark of AT&T.

[74] IMG image format, P. Raveling, is available via ftp/internet from raveling@venera.isi.edu.

[75] CompuServe Incorporated, Graphics Interchange Format(sm), "Programming Reference", Version 89a, (c)1987,1988,1989,1990, Columbus, Ohio, Document Date : 31 July 1990.

[76] TIFF image format, "An Aldus/Microsoft Technical Memorandum: 8/8/88"

This memorandum has been prepared jointly by Aldus and Microsoft in conjunction with leading scanner vendors and other interested parties. This document does not represent a commitment on the part of either Microsoft or Aldus to provide support for this file format in any application. When responding to specific issues raised in this memo, or when requesting additional tag or field assignments, please address your correspondence to either:

Developers_ Desk
Aldus Corporation
411 First Ave. South
Suite 200 Box 97017
Seattle, WA 98104
(206) 622-5500

Windows Marketing Group
Microsoft Corporation
16011 NE 36th Way
Redmond, WA 98073-9717
(206) 882-8080.

[77] Targa image file format, Appendix C of the Truevision Technical Guide. Requests for further information could be directed to:

Truevision, Inc.
7351 Shadeland Station, Suite 100
Indianapolis, IN 46256.

[78] PIXAR image format, "Pixar Image Computer ChapLibraries Users' Guide - Storage Formats", Version 2.2 November 1988, Copyright 1989, Pixar, 3240 Kerner Boulevard, San Rafael, CA 94901

[79] X11R4 image format, Copyright 1989, X Consortium, MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139.

[80] PPM image format, J. Poskanzer, is available via ftp/internet from jef@well.sf.ca.us.

[81] RLE image format, U. of Utah, is available via ftp/internet from cs.utah.edu.

[82] G. Buchsbaum, Color Signal Coding: Color Vision and Color Television, *Col. Res. Appl.*, **12**, 266-269 (1989).

[83] W. Ostwald, *Colour Science*, London: Winsor and Winsor (1931) and J. D. Foley and A. Van Dam, *Fundamental of Interactive Computer Graphics*,, 2nd ed., Reading MA: Addison-Wesley (1990).

[84] R. N. Bracewell, **The Fourier Transform and Its Applications**, 2 ed., McGraw-Hill, New York (1978).

[85] see Appendix A for a detailed description of the *crs* image format.

[86] A. H. Munsell, **A Color Notation**, 13 ed., Munsell Color, Baltimore (1979).

[87] F. Hurter and V. C. Driffield, Photochemical Investigations and a New Method of Determination of the Sensitiveness of Photographic Plates, *Jour. Soc. Chem. Ing.*, **9**, 455 (1890).

[88] B. M. Oliver, Tone Rendition in Television, *Proc. I. R.E.*, **38**, 1288-1300 (1950)

[89] G. Wyszecki, Correlate for Lightness in Terms of CIE Chromaticity Coordinates and Luminous Reflectance, *J. Opt. Soc. Am.*, **57**, 254-257 (1967).

[90] A. R. Robertson, The CIE 1976 Color-Difference Formulae, *Col. Res. Appl.*, **1**, 7-11 (1977).

[91] G. P. Corey, M. J. Clayton, and K. N. Cupery, Scene Dependence of Image Quality, *Phot. Sci. Eng.*, **27**, 9-13 (1983).

[92] Tristimulus Colorimeter C 1200, sn 0388992, LMT Lichtmesstechnik Berlin, Helmholtzstrasse 9, D 1000, Berlin 10.

[93] R. S. Berns and R. J. Motta, Colorimetric Calibration of Soft-Copy Devices to Aid in Hard-Copy Reproduction, *proceedings SPSE 41st annual conferance*, 266-269 (1988).

[94] R. S. Berns and M. E. Gorzynski, Neutral Reproduction of Soft-Copy and Hard-Copy Displays, AIC interrum meeting, June 1991, Syndey.

[95] M. D. Fairchild, A Novel Method for the Determination of Color Matching Functions, *Col. Res. Appl.*, **14**, 122-130 (1989).

[96] D. J. Finney, **Probit Analysis**, 3d ed., Cambridge U. Press, Cambridge, 1971.

[97] D. H. Alman, R. S. Berns, and W. A. Larsen, Performance Testing of Color-Difference Metrics Using a Color Tolerance Dataset,*Col. Res. Appl.*, (in press).

[98] SAS/STAT User's Guide, Version 6, Fourth Ed., 1071-1126 (1990).

[99]L. Reniff, Visual Determination of Color Differences using Probit Analysis: Phase II, Master's Thesis, Rochester Institute of Technology (1989).

[100] D. Pregibon, Logistic Regression Diagnostics, *Annals of Stat.*, **9**, 705-724 (1981).

[101] M. R. Luo and B. Rigg, BFD (l:c) Colour-Difference Formula, Part I Development of the Formula, *J. Soc. Dyers Col.* **103**, 86-132 (1987).

[102] S. Stamm, An Investigation of Color Tolerance,*Proc. Tech. Assoc. Graphic Arts*, 156-173 (1981).

[103] Steve Viggianno, personal communication.

[104] J. von Kries, Uber die anomalen trichormatische farbensysteme, *Z. Sinnephysiol.*, **19**, 63-69 (1899).

# Appendix A : CRS Storage Format

```
/* CRS External Header (disk file) ***********************************/

typedef struct {
 char storage_format[9];   /* file storage format */
 char header_size[5];      /* size of header */
 char atx[5];              /* x origin */
 char aty[5];              /* y origin */
 char image_width[5];      /* image width in pixels */
 char image_height[5];     /* image height in pixels */
 char image_linelength[6]; /* image line length */
 char image_length[9];     /* image length */
 char color_planes[2];     /* number of color planes */
 char color_bits[3];       /* number of bits per pixel */
 char crs_flag[3];         /* general flag *
                           /* 1: draft/final mode */
                           /* 2: rle compression */
                           /* byte order */
                           /* center image */
                           /* type of origin (lo left vs. up left) */
 char cst_flag[3];         /* color space transformation flag */
                           /* 0 : BAD_SPACE */
                           /* 1 : crt_SPACE */
                           /* 2 : Yxy_SPACE */
                           /* 3 : XYZ_SPACE */
                           /* 4 : Lab_SPACE */
                           /* 5 : LabLCh_SPACE */
                           /* 6 : Luv_SPACE */
                 /         * 7 : LuvLCh_SPACE */
 char ccf_flag[3];         /* color corrections flag */
 char author[49];          /* author from passwd entry */
 char date[33];            /* Unix format date */
 char program[33];         /* program that created image */
 char wp_X[8];             /* white point for tristimulus value : X */
 char wp_Y[8];             /* white point for Y */
 char wp_Z[8];             /* white point for Z */
 char min_1[8];            /* minimum value for channel 1 */
 char min_2[8];            /* minimum value for channel 2 */
 char min_3[8];            /* minimum value for channel 3 */
 char max_1[8];            /* maximum value for channel 1 */
 char max_2[8];            /* maximum value for channel 2 */
 char max_3[8];            /* maximum value for channel 3 */
 char mean_1[8];           /* mean value for channel 1 */
```

```
 char mean_2[8];          /* mean value for channel 2 */
 char mean_3[8];          /* mean value for channel 3 */
 char anchor_1[8];        /* anchor value for channel 1 */
 char anchor_2[8];        /* anchor value for channel 2 */
 char anchor_3[8];        /* anchor value for channel 3 */
 char comments[129];      /* comments */
 } CRS_ext_hdr;

/* CRS Internal Header (RAM memory)  ******************************/

typedef struct {
 int atx;                 /* x origin */
 int aty;                 /* y origin */
 int image_width;         /* image width in pixels */
 int image_height;        /* image height in pixels */
 int image_linelength;    /* image line length */
 int image_length;        /* image length */
 int color_planes;        /* number of color planes */
 int color_bits;          /* number of bits per pixel */
 int crs_flag;            /* crs general flags */
 int cst_flag;            /* color space transformation flag */
 int ccf_flag;            /* color corrections flag */
 char *author;            /* author from passwd entry */
 char *date;              /* Unix format date */
 char *program;           /* program that created image */
 float wp[3];             /* white point for tristimulus values */
 float min[3];            /* minimum values for channels */
 float max[3];            /* maximum values for channels */
 float mean[3];           /* mean values for channels */
 float anchor[3];         /* anchor values for channels */
 char *comments;          /* comments */
 } CRS_int_hdr;

typedef struct {
  CRS_int_hdr hdr;
  float *data; /* all internal calculations performed in floating point !! */
 } crsfs;
```

# Appendix B : Transfer Function Parameter Levels

| Lightness | | Chroma | | Hue Angle | |
|---|---|---|---|---|---|
| Function | Level | Function | Level | Function | Level |
| MFACTOR | 0.975 | MFACTOR | 0.967 | VOFFSET | -20.0 |
| MFACTOR | 0.95 | MFACTOR | 0.933 | VOFFSET | -15.0 |
| MFACTOR | 0.925 | MFACTOR | 0.90 | VOFFSET | -12.5 |
| MFACTOR | 0.90 | MFACTOR | 0.85 | VOFFSET | -10.0 |
| MFACTOR | 0.875 | MFACTOR | 0.80 | VOFFSET | -7.5 |
| MFACTOR | 0.85 | MFACTOR | 0.70 | VOFFSET | -5.0 |
| POWER | 1.30 | POWER | 1.30 | VOFFSET | -2.5 |
| POWER | 1.25 | POWER | 1.25 | VOFFSET | 1.67 |
| POWER | 1.20 | POWER | 1.20 | VOFFSET | 3.34 |
| POWER | 1.15 | POWER | 1.15 | VOFFSET | 5.0 |
| POWER | 1.10 | POWER | 1.125 | VOFFSET | 7.5 |
| POWER | 1.05 | POWER | 1.10 | VOFFSET | 10.0 |
| POWER | 0.96 | POWER | 1.075 | VOFFSET | 12.5 |
| POWER | 0.92 | POWER | 1.05 | VOFFSET | 15.0 |
| POWER | 0.88 | POWER | 0.967 | | |
| POWER | 0.84 | POWER | 0.933 | | |
| POWER | 0.80 | POWER | 0.90 | | |
| POWER | 0.76 | POWER | 0.85 | | |
| SCURVE | 1.55 | POWER | 0.80 | | |
| SCURVE | 1.40 | POWER | 0.70 | | |
| SCURVE | 1.35 | POWER | 0.60 | | |
| SCURVE | 1.30 | | | | |
| SCURVE | 1.24 | | | | |
| SCURVE | 1.21 | | | | |
| SCURVE | 1.17 | | | | |
| SCURVE | 1.13 | | | | |
| SCURVE | 1.09 | | | | |
| SCURVE | 1.05 | | | | |
| SCURVE | 0.96 | | | | |
| SCURVE | 0.92 | | | | |
| SCURVE | 0.88 | | | | |
| SCURVE | 0.84 | | | | |
| SCURVE | 0.80 | | | | |
| SCURVE | 0.76 | | | | |

# Appendix C : Pilot Experiment

A pilot experiment was performed in order to verify the experimental design, clarify the instructions, and to determine proper parameter levels for the transfer functions. The pilot experiment was performed using the same environment as the full experiment. Eleven color normal observers from the Color Science department judged two images that had been manipulated using the same transfer functions and color space dimension combinations as the full experiment. The total experiment was completed in one one-hour observational session for each observer. The results are show below. The excellent Chi-Squared results were unexpected and encouraging. Of interest is the poor performance in positive hue angle offset and the lightness high sigmoidal function. This was due to the maximum parameter levels not being large enough and the increments being too large. The most interesting result of the pilot is the complete agreement of the tolerances to those in the final experiment. In essence, the only advantage of using more images and observers was tighter confidence intervals.

Pilot Experiment Perceptibility Results

| Function | $Chi^2$ | $Pr > Chi^2$ | T50 | LOWER | UPPER |
|---|---|---|---|---|---|
| LMF | 6.4196 | 0.4917 | 0.93839 | 0.92555 | 0.95638 |
| LPH | 2.6969 | 0.4408 | 1.08161 | 1.03103 | 1.11382 |
| LPL | 2.2301 | 0.5260 | 0.92568 | 0.89479 | 0.97892 |
| LSH | 1.0524 | 0.7886 | 1.14775 | . | . |
| LSL | 1.8724 | 0.5993 | 0.92024 | 0.88923 | 0.97603 |
| CMF | 10.1308 | 0.1813 | 0.90627 | 0.87623 | 0.96528 |
| CPL | 3.1866 | 0.3637 | 1.10043 | 1.05569 | 1.12405 |
| CPH | 5.3116 | 0.1504 | 0.88402 | 0.85627 | 0.93158 |
| HOH | 1.6364 | 0.6512 | 4.7664 | . | . |
| HOL | 5.0538 | 0.1679 | -8.5806 | -11.6605 | -4.3236 |

Pilot Experiment Acceptability Results

| Function | $Chi^2$ | $Pr > Chi^2$ | T50 | High | Low |
|---|---|---|---|---|---|
| LMF | 2.1988 | 0.9480 | 0.90506 | 0.89578 | 0.91402 |
| LPH | 0.6219 | 0.8914 | 1.22495 | 1.19463 | 1.32496 |
| LPL | 0.5189 | 0.9147 | 0.85534 | 0.83135 | 0.87483 |
| LSH | 4.3208 | 0.2288 | 1.36577 | 1.24581 | 5.71915 |
| LSL | 2.7102 | 0.4385 | 0.83214 | 0.79759 | 0.85493 |
| CMF | 6.1487 | 0.5225 | 0.84116 | 0.82287 | 0.86002 |
| CPL | 1.0202 | 0.7964 | 1.15486 | 1.13468 | 1.18197 |
| CPH | 0.4851 | 0.9222 | 0.79149 | 0.73417 | 0.81182 |
| HOH | 9.9704 | 0.0188 | 9.8168 | -1.7469 | 18.1579 |
| HOL | 4.9353 | 0.1766 | -12.7801 | -14.7810 | -10.9210 |

# Appendix D : Monitor Calibration SAS Code

## Linearization Code (RGB dac Values => RGB Tristimulus Values)

```
options nocenter device=proprint linesize=78 pagesize=60;
libname here "d:\sas\mds\hw2\";

/* 1 : read in monitor ramp data */
data here.gammacal;
 infile "gammacal.dat" firstobs = 2;
 input dac R G B;
 dac = dac / 255;
 run;

/* 2 : plot monitor response */
symbol1 i=spline v=none;
proc gplot data=here.gammacal;
 plot dac*R=1 dac*G=1 dac*B=1 / overlay;
 run;

/* 3 : model nonlinear portion of monitor (dac <=> linear RGB) */
proc model data=here.gammacal;
 R = (roff + rgain*dac)**rgamma;
 G = (goff + ggain*dac)**ggamma;
 B = (boff + bgain*dac)**bgamma;

 label roff='red offset'
       rgain='red gain'
       rgamma='red gamma'
       goff='green offset'
       ggain='green gain'
       ggamma='green gamma'
       boff='blue offset'
       bgain='blue gain'
       bgamma='blue gamma';

 fit R G B start= (roff 0 rgain 1 rgamma 2.3
                   goff 0 ggain 1 ggamma 2.3
                   boff 0 bgain 1 bgamma 2.3
          ) / outest=here.rgbmodel;
 run;

/* 4 : print regression results */
```

```
proc print data=here.rgbmodel;
  title "Nonlinear CRT model Parameters";
  run;
```

## Conversion Code
## (RGB Tristimulus Values <=> XYZ Tristimulus Values)

```
options linesize=80 pagesize=60;
proc iml;

/* White Point Values */
pixar = {       23.16 14.79 8.7,
                12.86 32.33 4.12,
                1.30 6.64 45.85};

/* Normalize such that Y max = 100.0 */
yp = sum(pixar(|2,|))/100.0;

/* RGB => XYZ */
fromp = pixar / yp;

/* XYZ => RGB */
top = inv(fromp);

print fromp top;
quit;
```

# Appendix E : Survey Results

The survey below was given to each observer after completion of the final observational session. The summarized responses follow the survey form.

**Observational Survey**

Name : _____ Age : _____ years old

Please take a few minutes to fill out the following questionnaire. There are no right or wrong answers. Criticism is welcome !

1. How much previous experience have you had in judging images

2. What did you think of the instructions ?

3. What were your general impressions of the experiment ?

4. Did you find yourself making judgements in any particular pattern, method, or with any particular criteria ?

5. What did you think about the length of the experiment and the sessions ?

6. Did you have any problems with any of the image(s) ?

7. Would you be willing to be an observer again ? If not, why not ?

8. Do you know anyone (that was not an observer) who would be willing to be an observer ?

The average observer age was 30 years old with a standard deviation of 7.4 years. The youngest was 20 and the oldest was 49.

1. 75 % had little or no experience judging images
   15 % had moderate experience judging images
   10 % had judged images professionally for several years

2. 85 % thought the instructions were clear and complete
   15 % thought the instructions were somewhat vague or confusing

3. 35 % thought the experiment was "fun" or "very good"
   15 % thought the experiment was good
   15 % thought the experiment was too long
   35 % had specific comments about particular parts of the experiment such as the reference images were of poor quality

4. 45 % selected particular objects in the images for reference
   25 % used whites as a reference
   20 % used no method
   10 % judged general color shifts

   Some observers commented at this point that they ignored the acceptability criteria and used their own acceptability criteria. This accounting for some of the noise in the acceptability results.

5. 80 % thought the length of the experiment was good or about right
   20 % thought the length of the experiment was too long

6. 37 % had no difficulty with any particular image
   33 % had difficulty with image #1 of the woman (and commented about the lack of colors to judge)
   20 % had difficulty with image #6 of the leaves
   10 % had difficulty with image #2 of the dessert

7. 90 % were willing to be observers again
   10 % were not willing to be observers again

8. 50 % knew of someone else who would be willing to be an observer
   50 % did not know anyone who would be willing to be an observer

The overwhelming response to questions seven and eight indicate unusually good experimental design from the observer's point of view.

# Appendix F : SAS Code used for Results

## Make SAS data sets

```
options pagesize=66 linesize=80;
libname hdd "user18:[mds3399.thesis.experiment]";

filename imgfile "imglist.all";

data hdd.lmf hdd.lph hdd.lpl hdd.lsh hdd.lsl
    hdd.cmf hdd.cph hdd.cpl hdd.hoh hdd.hol;
  infile imgfile;
  input stdfile $char11. smpfile $char11.
        lshaper $ lvalue cshaper $ cvalue hshaper $ hvalue
      passfail $ stdsmp $ hitcount time initials $;

  if (lshaper = 'MFACTOR') then output hdd.lmf;
  if (lshaper = 'POWER' & lvalue >= 1.0) then output hdd.lph;
  if (lshaper = 'POWER' & lvalue <= 1.0) then output hdd.lpl;
  if (lshaper = 'SCURVE' & lvalue >= 1.0) then output hdd.lsh;
  if (lshaper = 'SCURVE' & lvalue <= 1.0) then output hdd.lsl;
  if (cshaper = 'MFACTOR') then output hdd.cmf;
  if (cshaper = 'POWER' & cvalue >= 1.0) then output hdd.cph;
  if (cshaper = 'POWER' & cvalue <= 1.0) then output hdd.cpl;
  if (hshaper = 'VOFFSET' & hvalue >= 0.0) then output hdd.hoh;
  if (hshaper = 'VOFFSET' & hvalue <= 0.0) then output hdd.hol;

  run;
```

## Filtered Perceptibility Results

```
options nonumber nocenter pagesize=66 linesize=80;

libname hdd "user18:[mds3399.thesis.experiment.flt]";

%macro doprobit(dset, parm1, parm2, lmt1, lmt2);
  proc logistic data=hdd.&dset nosimple noprint;
    model stdsmp  = &parm1 / link=normit;
    output out=a &parm2=&parm2;
    run;

 data &dset;
   set a;
    if ((&parm2 <= &lmt1) or (&parm2 >= &lmt2)) then delete;
```

```
  run;

 proc probit data=&dset c=0.5 inversecl lackfit;
   title "Filtered Perceptibility Data : &dset, &parm1, &parm2, &lmt1, &lmt2";
  class stdsmp;
   model stdsmp = &parm1 / d=normal;
   run;

%mend doprobit;

%doprobit(lmf, lvalue, c, -0.01, 0.01);
%doprobit(lph, lvalue, c, -0.01, 0.01);
%doprobit(lpl, lvalue, c, -0.01, 0.01);
%doprobit(lsh, lvalue, c, -0.01, 0.01);
%doprobit(lsl, lvalue, c, -0.01, 0.01);
%doprobit(cmf, cvalue, c, -0.01, 0.01);
%doprobit(cph, cvalue, c, -0.01, 0.01);
%doprobit(cpl, cvalue, c, -0.01, 0.01);
%doprobit(hoh, hvalue, c, -0.01, 0.01);
%doprobit(hol, hvalue, c, -0.01, 0.01);
```

## Filtered Perceptibility Results By Scene

```
options nonumber nocenter pagesize=66 linesize=80;

libname hdd "user18:[mds3399.thesis.experiment.flt]";

%macro doprobit(dset, parm1, parm2, lmt1, lmt2);
 proc logistic data=hdd.&dset nosimple noprint;
   model stdsmp  = &parm1 / link=normit;
   output out=a &parm2=&parm2;
   run;

 data &dset;
  set a;
   if ((&parm2 <= &lmt1) or (&parm2 >= &lmt2)) then delete;
   run;

 proc sort data=&dset;
  by stdfile;
   run;

 proc probit data=&dset c=0.5 inversecl lackfit;
  by stdfile;
```

```
    title "Filtered Perceptibility Data : By Scene : &dset, &parm1, &parm2,
&lmt1, &lmt2";
   class stdsmp;
    model stdsmp = &parm1 / d=normal;
    run;

%mend doprobit;

%doprobit(lmf, lvalue, c, -0.01, 0.01);
%doprobit(lph, lvalue, c, -0.01, 0.01);
%doprobit(lpl, lvalue, c, -0.01, 0.01);
%doprobit(lsh, lvalue, c, -0.01, 0.01);
%doprobit(lsl, lvalue, c, -0.01, 0.01);
%doprobit(cmf, cvalue, c, -0.01, 0.01);
%doprobit(cph, cvalue, c, -0.01, 0.01);
%doprobit(cpl, cvalue, c, -0.01, 0.01);
%doprobit(hoh, hvalue, c, -0.01, 0.01);
%doprobit(hol, hvalue, c, -0.01, 0.01);
```

## Raw Acceptability Results

```
options nonumber nocenter pagesize=66 linesize=80;

libname hdd "user18:[mds3399.thesis.experiment]";

%macro doprobit(dset, parm1);
  proc probit data=hdd.&dset inversecl lackfit;
    title "Raw Acceptability Data : &dset : ";
    class passfail;
    model passfail = &parm1 / d=normal;
    run;
%mend doprobit;

%doprobit(lmf, lvalue);
%doprobit(lph, lvalue);
%doprobit(lpl, lvalue);
%doprobit(lsh, lvalue);
%doprobit(lsl, lvalue);
%doprobit(cmf, cvalue);
%doprobit(cph, cvalue);
%doprobit(cpl, cvalue);
%doprobit(hoh, hvalue);
%doprobit(hol, hvalue);
```

## Raw Acceptability Results By Scene

```
options nonumber nocenter pagesize=66 linesize=80;

libname hdd "user18:[mds3399.thesis.experiment]";

%macro doprobit(dset, parm1);
 proc sort data=hdd.&dset;
  by stdfile;
  run;

 proc probit data=hdd.&dset inversecl lackfit;
  by stdfile;
   title "Raw Acceptability Data : By Scene : &dset";
  class passfail;
   model passfail = &parm1 / d=normal;
   run;
%mend doprobit;

%doprobit(lmf, lvalue);
%doprobit(lph, lvalue);
%doprobit(lpl, lvalue);
%doprobit(lsh, lvalue);
%doprobit(lsl, lvalue);
%doprobit(cmf, cvalue);
%doprobit(cph, cvalue);
%doprobit(cpl, cvalue);
%doprobit(hoh, hvalue);
%doprobit(hol, hvalue);
```

# Appendix G : C Code Library Manual Pages



**NAME**

libmisc -crs routines to perform miscellaneous support functions

**DESCRIPTION**

The routines in this library provide error reporting, active window control, image struction initialization, image structure parameter input and image struction conversion routines for the *crs* software environment.

**FUNCTION LIST**

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| crs_warn | crs_error(3) | -issues a warning and returns to the calling routine |
| crs_error | crs_error(3) | -issues a warning to stderr and exits to shell |
| load_actwin | actwin(3) | -loads parameters into active window structure |
| reset_actwin | actwin(3) | -resets active window structure |
| copy_crshdr | copy_crshdr(3) | -makes a copy of a crs format header |
| init_crs | init_crs(3) | -initializes a crs header structure |
| init_mds | init_crs(3) | -initializes an mds header structure |
| init_pix | init_crs(3) | -initializes a pixar header structure |
| init_ppm | init_crs(3) | -initializes a ppm header structure |
| shin_crshdr | shin_hdr(3) | -loads parameters into a crs format header structure |
| shin_crthdr | shin_hdr(3) | -loads parameters into a crt format header structure |
| shin_mdshdr | shin_hdr(3) | -loads parameters into a mds format header structure |
| shin_ppmhdr | shin_hdr(3) | -loads parameters into a ppm format header structure |
| shin_xwdhdr | shin_hdr(3) | -loads parameters into a xwd format header structure |
| crs2crt | crs_convsersions(3) | -converts crs image structure into crt image structure |
| crs2mds | crs_convsersions(3) | -converts crs image structure into mds image structure |
| crs2pix | crs_convsersions(3) | -converts crs image structure into pix image structure |
| crs2ppm | crs_convsersions(3) | -converts crs image structure into ppm image structure |
| crs2xwd | crs_convsersions(3) | -converts crs image structure into xwd image structure |
| crt2crs | crs_convsersions(3) | -converts crt image structure into crs image structure |
| mds2crs | crs_convsersions(3) | -converts mds image structure into crs image structure |

| | | |
|---|---|---|
| mds2pix | crs_convsersions(3) | -converts mds image structure into ppm image structure |
| mds2ppm | crs_convsersions(3) | -converts mds image structure into pixar image structure |
| mds2xwd | crs_convsersions(3) | -converts mds image structure into xwd image structure |
| pix2crs | crs_convsersions(3) | -converts pixar image structure into crs image structure |
| ppm2crs | crs_convsersions(3) | -converts ppm image structure into crs image structure |
| ppm2mds | crs_convsersions(3) | -converts ppm image structure into mds image structure |
| ppm2xwd | crs_convsersions(3) | -converts ppm image structure into xwd image structure |
| xwd2mds | crs_convsersions(3) | -converts xwd image structure into mds image structure |
| xwd2ppm | crs_convsersions(3) | -converts xwd image structure into ppm image structure |
| readnextrow | nextrow(3) | -reads next row of pixels |
| writenextrow | nextrow(3) | -writes next row of pixels |
| load_minmax | load_minmax(3) | -loads minimum and maximum parameters into a crs |

**FILES**

/usr/local/lib/libcrs.a crs runtime library

**SEE ALSO**

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

**crs_warn** -issues a warning and returns to the calling routine

**crs_error** -issues a warning to stderr and exits to shell

## SYNOPSIS

```
#include <crs.h>

void crs_warn(message)
char *message;          -descriptive warning message

void crs_error(message)
char *message;          -descriptive error message
```

## DESCRIPTION

These routines provide simple error reporting support.

## FILES

/usr/local/lib/libcrs.a

## NAME

**load_actwin** -loads parameters into active window structure
**reset_actwin** -resets active window structure

## SYNOPSIS

```
#include <crs.h>

int load_actwin(win, x_min, y_min, x_max, y_max)
actwin_ptr win;
int x_min, y_min, x_max, y_max;

int reset_actwin(crs, win,)
crsimg_ptr crs;
actwin_ptr win;
```

## DESCRIPTION

An active window is the rectangular area of an image on which other operations take place. This concept provides the ability to select an area of interest in the image for various manipulations without having to operate on the entire image.

## FILES

/usr/local/lib/libcrs.a

## NAME

**copy_crshdr** -makes a copy of a crs format header

## SYNOPSIS

```
#include <crs.h>

int copy _crshdr(orig, dupe, win)
crsimg_ptr orig, dupe;
actwin_ptr win;
```

## DESCRIPTION

This routine simply copies the header structure of a crs image. This is useful when comparing original and manipulated images.

## FILES

/usr/local/lib/libcrs.a

## NAME

| | |
|---|---|
| **init_crs** | -initializes a crs header structure |
| **init_mds** | -initializes a mds header structure |
| **init_pix** | -initializes a pix header structure |
| **init_ppm** | -initializes a ppm header structure |

## SYNOPSIS

```
#include <crs.h>

int init_crs(crs)
crsimg_ptr crs;

int init_mds(mds)
mdsimg_ptr mds;

int init_pix(pix)
piximg_ptr pix;

int init_ppm(ppm)
ppmimg_ptr ppm;
```

## DESCRIPTION

These routines initialize various image structures to a default state and free up any dynamically allocated memory within the structure.

## FILES

/usr/local/lib/libcrs.a

## NAME

**shin_crshdr** -loads parameters into a crs format header structure

**shin_crthdr** -loads parameters into a crtformat header structure

**shin_mdshdr** -loads parameters into a mds format header structure

**shin_ppmhdr** -loads parameters into a ppm format header structure

**shin_xwdhdr** -loads parameters into a xwd format header structure

## SYNOPSIS

```
#include <crs.h>

int shin_crshdr(crs, atx, aty, image_width, image_height,
                image_linelength, image_length, color_planes, color_bits,
                crs_flag, cst_flag, ccf_flag, *author, *date, *program, wp_X, wp_Y,
wp_Z
                min_1, min_2, min_3, max_1, max_2, max_3,
                mean_1, mean_2, mean_3, anchor_1, anchor_2, anchor_3,*comments)
crsimg_ptr crs;
int atx;                              -x origin
int aty;                              -y origin
int image_width;                      -image width in pixels
int image_height;                     -image height in pixels
int image_linelength;                 -image line length
int image_length;                     -image length
int color_planes;                     -number of color planes
int color_bits;                       -number of bits per pixel
int crs_flag;                         -crs general flags
int cst_flag;                         -color space transformation flag
int ccf_flag;                         -color corrections flag
char *author;                         -author from passwd entry
char *date;                           -Unix format date
char *program;                        -program that created image
float wp_X, wp_Y, wp_Z;               -white point for tristimulus values
float min_1, min_2, min_3;            -minimum values for channels
float max_1, max_2, max_3;            -maximum values for channels
float mean_1, mean_2, mean_3;         -mean values for channels
float anchor_1, anchor_2, anchor_3;   -anchor values for channels
char *comments;                       -comments

int shin_crthdr(crt, format, atx, aty, image_width, image_height, color_planes)
crtimg_ptr crt;
int format;                           -format code
int atx;                              -x origin
int aty;                              -y origin
```

```
int image_width;                    -image width in pixels
int image_height;                   -image height in pixels
int color_planes;                   -number of color planes

int shin_mdshdr(mds, format, image_width, image_height,
                image_depth, image_linelength, image_length)
mdsimg_ptr mds;
int format;                         -format code
int image_width;                    -image width in pixels
int image_height;                   -image height in pixels
int image_depth;                    -image depth
int image_linelength;               -image line length
int image_length;                   -image length

int shin_ppmhdr(ppm, image_width, image_height, max_val)
ppmmg_ptr ppm;
int image_width;                    -image width in pixels
int image_height;                   -image height in pixels
int max_val;                        -maximum pixel value


int shin_xwdhdr(xwd, header_size,  file_version, pixmap_format,
                pixmap_depth, pixmap_width, pixmap_height, xoffset,
                byte_order, bitmap_unit, bitmap_bit_order, bitmap_pad,
                bits_per_pixel, bytes_per_line, visual_class,
                red_mask, green_mask, blue_mask, bits_per_rgb,
                colormap_entries, ncolors,
                window_width, window_height, window_x, window_y,
                window_bdrwidth)
xwdimg_ptr xwd;
xwdval header_size;                 -Size of file header (bytes).
xwdval file_version;                -XWD_FILE_VERSION
xwdval pixmap_format;               -Pixmap format
xwdval pixmap_depth;                -Pixmap depth
xwdval pixmap_width;                -Pixmap width
xwdval pixmap_height;               -Pixmap height
xwdval xoffset;                     -Bitmap x offset
xwdval byte_order;                  -MSBFirst, LSBFirst
xwdval bitmap_unit;                 -Bitmap unit
xwdval bitmap_bit_order;            -MSBFirst, LSBFirst
xwdval bitmap_pad;                  -Bitmap scanline pad
xwdval bits_per_pixel;              -Bits per pixel
xwdval bytes_per_line;              -Bytes per scanline
xwdval visual_class;                -Class of colormap
xwdval red_mask;                    -Z red mask
xwdval green_mask;                  -Z green mask
xwdval blue_mask;                   -Z blue mask
xwdval bits_per_rgb;                -Log base 2 distinct color values
xwdval colormap_entries;            -Number of entries in colormap
xwdval ncolors;                     -Number of Color structures
xwdval window_width;                -Window width
xwdval window_height;               -Window height
long window_x;                      -Window upper left X coordinate
long window_y;                      -Window upper left Y coordinate
```

**xwdval window_bdrwidth;** -Window border width

## DESCRIPTION

These routines provide a straightforward method for defining parameters in various image structures. These parameters must be defined before the image can be manipulated.

## FILES

/usr/local/lib/libcrs.a

## NAME

**crs2crt** -converts crs image structure into crt image structure

**crs2mds** -converts crs image structure into mds image structure

**crs2pix** -converts crs image structure into pixar image structure

**crs2ppm** -converts crs image structure into ppm image structure

**crs2xwd** -converts crs image structure into xwd image structure

**crt2crs** -converts crt image structure into crs image structure

**mds2crs** -converts mds image structure into crs image structure

**mds2pix** -converts mds image structure into pixar image structure

**mds2ppm** -converts mds image structure into ppm image structure

**mds2xwd** -converts mds image structure into xwd image structure

**pix2crs** -converts pixar image structure into crs image structure

**pix2mds** -converts pixar image structure into mds image structure

**pix2ppm** -converts pixar image structure into ppm image structure

**ppm2crs** -converts ppm image structure into crs image structure

**ppm2mds** -converts ppm image structure into mds image structure

**ppm2pix** -converts ppm image structure into pixar image structure

**ppm2xwd** -converts ppm image structure into xwd image structure

**xwd2mds** -converts xwd image structure into mds image structure

**xwd2ppm** -converts xwd image structure into ppm image structure

## SYNOPSIS

```
#include <crs.h>

int crs2crt(crs, crt, win, crtmodel crt3x3)
crsimg_ptr crs;
crtimg_ptr crt;
actwin_ptr win;
shaper_ptr crtmodel;
```

```
shaper_ptr crt3x3;

int crs2mds(crs, mds win, crtmodel crt3x3)
crsimg_ptr crs;
mdsimg_ptr mds;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int crs2pix(crs, pix, win, crtmodel crt3x3)
crsimg_ptr crs;
piximg_ptr pix;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int crs2ppm(crs, ppm, win, crtmodel crt3x3)
crsimg_ptr crs;
ppmimg_ptr ppm;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int crs2xwd(crs, xwd win, crtmodel crt3x3)
crsimg_ptr crs;
xwdimg_ptr xwd;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int crt2crs(crt, crs, win, crtmodel crt3x3)
crtimg_ptr crt;
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int mds2crs(mds, crs, win, crtmodel crt3x3)
mdsimg_ptr mds;
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int mds2pix(mds, pix)
mdsimg_ptr mds;
piximg_ptr pix;

int mds2ppm(mds, ppm)
mdsimg_ptr mds;
ppmimg_ptr ppm;

int mds2xwd(mds, xwd)
mdsimg_ptr mds;
```

```
xwdimg_ptr xwd;

int pix2crs(pix, crs, win, crtmodel crt3x3)
piximg_ptr pix;
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int pix2mds(pix, mds)
piximg_ptr pix;
mdsimg_ptr mds;

int pix2ppm(pix, ppm)
piximg_ptr pix;
ppmimg_ptr mds;

int ppm2crs(ppm, crs, win, crtmodel crt3x3)
ppmimg_ptr ppm;
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int ppm2pix(ppm, pix)
ppmimg_ptr ppm;
piximg_ptr pix;

int ppm2mds(ppm, mds)
ppmimg_ptr ppm;
mdsimg_ptr mds;

int ppm2xwd(ppm, xwd)
ppmimg_ptr ppm;
xwdimg_ptr xwd;

int xwd2mds(xwd, mds)
xwdimg_ptr xwd;
mdsimg_ptr mds;

int xwd2ppm(xwd, ppm)
xwdimg_ptr xwd;
ppmimg_ptr ppm;
```

DESCRIPTION

These routines provide conversion utilities between various image structures.

FILES

/usr/local/lib/libcrs.a

## NAME

| | |
|---|---|
| **read_nextrow** | -reads next row of pixels |
| **write_nextrow** | -writes next row of pixels |

## SYNOPSIS

```
#include <crs.h>

int read_nextrow(from, nextrow, row, columns)
float *from;
float *nextrow;
int row;
int columns;

int write_nextrow(to, nextrow, row, columns)
float *to;
float *nextrow;
int row;
int columns;
```

## DESCRIPTION

These routines are internal support routines for the crs_scale routine for scaling images.

## FILES

/usr/local/lib/libcrs.a

## NAME

**load_minmax** -loads minimum and maximum parameters into a crs

## SYNOPSIS

```
#include <crs.h>

int load_minmax(crs, min_1, min_2, min_3, max_1, max_2, max_3)
crsimg_ptr crs;
float min_1, min_2, min_3;
float max_1, max_2, max_3;
```

## DESCRIPTION

This routine loads the minimum and maximum values for the three dimensions of the image's color space. These values are required for several operations such a power functions, which use such anchor points.

## FILES

/usr/local/lib/libcrs.a

**NAME**

libcst -crs routines to perform color space transformations

**DESCRIPTION**

This library provides the basic routines for convert images and pixels from one color space to another. The crs_cst routine operates on an entire crs image, calling the necessary pixel operations.

**FUNCTION LIST**

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| crs_cst | crs_cst(3) | -converts crs image between color spaces |
| Lab_LabLCh pixel | crs_cst(3) | -converts a CIELAB pixel into a CIELAB L*C*h° |
| Lab_XYZ | crs_cst(3) | -converts a CIELAB pixel into a tristimulus pixel |
| LabLCh_Lab pixel | crs_cst(3) | -converts a CIELAB L*C*h° pixel into a CIELAB |
| Luv_LuvLCh pixel | crs_cst(3) | -converts a CIELUV pixel into a CIELUV L*C*h° |
| Luv_XYZ | crs_cst(3) | -converts a CIELUV pixel into a tristimulus pixel |
| LuvLCh_Luv pixel | crs_cst(3) | -converts a CIELUV L*C*h° pixel into a CIELUV |
| scn_XYZ tristimulus pixel | crs_cst(3) | -converts a raw rgb scanner pixel into a |
| XYZ_crt | crs_cst(3) | -converts a tristimulus pixel into a CRT rgb pixel |
| XYZ_Lab | crs_cst(3) | -converts a tristimulus pixel into a CIELAB pixel |
| XYZ_Ljg | crs_cst(3) | -converts a tristimulus pixel into an OSA Ljg pixel |
| XYZ_Luv | crs_cst(3) | -converts a tristimulus pixel into a CIELUV pixel |
| XYZ_YIQ pixel | crs_cst(3) | -converts a tristimulus pixel into a NTSC YIQ |
| XYZ_Yxy pixel | crs_cst(3) | -converts a tristimulus pixel into a chromaticity |
| YIQ_XYZ tristimulus | crs_cst(3) | -converts a NTSC YIQ pixel into a pixel |
| Ljg_XYZ pixel | crs_cst(3) | -converts an OSA Ljg pixel into a tristimulus |
| Yxy_XYZ pixel | crs_cst(3) | -converts a chromaticity pixel into a tristimulus |

**FILES**

/usr/local/lib/libcrs.a crs runtime library

**SEE ALSO**

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

**crs_cst** -performs color space conversions on crs images
**crt_hls** -converts a CRT rgb pixel into an hls pixel
**crt_XYZ** -converts a CRT rgb pixel into a tristimulus pixel
**hls_crt** -converts an hls rgb pixel into a CRT pixel
**Lab_LabLCh** -converts a CIELAB pixel into a CIELAB L*C*h° pixel
**Lab_XYZ** -converts a CIELAB pixel into a tristimulus pixel
**LabLCh_Lab** -converts a CIELAB L*C*h° pixel into a CIELAB pixel
**Luv_LuvLCh** -converts a CIELUV pixel into a CIELUV L*C*h° pixel
**Luv_XYZ** -converts a CIELUV pixel into a tristimulus pixel
**LuvLCh_Luv** -converts a CIELUV L*C*h° pixel into a CIELUV pixel
**scn_XYZ** -converts a raw rgb scanner pixel into a tristimulus pixel
**XYZ_crt** -converts a tristimulus pixel into a CRT rgb pixel
**XYZ_Lab** -converts a tristimulus pixel into a CIELAB pixel
**XYZ_Ljg** -converts a tristimulus pixel into an OSA Ljg pixel
**XYZ_Luv** -converts a tristimulus pixel into a CIELUV pixel
**XYZ_YIQ** -converts a tristimulus pixel into a NTSC YIQ pixel
**XYZ_Yxy** -converts a tristimulus pixel into a chromaticity pixel
**YIQ_XYZ** -converts a NTSC YIQ pixel into a pixel tristimulus
**Ljg_XYZ** -converts an OSA Ljg pixel into a tristimulus pixel
**Yxy_Ljg** -converts a chromaticity pixel into an OSA Ljg pixel
**Yxy_XYZ** -converts a chromaticity pixel into a tristimulus pixel

## SYNOPSIS

```
#include <crs.h>

int crs_cst(from_space, to_space, from_crs, to_crs,
        win, crtmodel, crt3x3)
int from_space;          -original color space
int to_space;            -destination color space
crsimg_ptr from_crs;     -original crs image
crsimg_ptr to_crs;       -destination crs image
actwin_ptr win;          -active window structure
shaper_ptr crtmodel;     -crt model structure
shaper_ptr crt3x3;       -crt 3x3 structure

int crt_hls(crt, hls, levels)
```

```
float crt[3];                          -crt pixel
float hls[3];                          -hls pixel
float levels;                          -number of digital count levels

int crt_XYZ(crt, XYZ, levels, crtmodel, crt3x3)
int crt[3];                            -CRT pixel
float XYZ[3];                          -XYZ pixel
float levels;                          -number of digital count levels
shaper_ptr crtmodel;                   -crt model structure
shaper_ptr crt3x3;                     -crt 3x3 structure

int hls_crt(hls, crt, levels)
float hls[3];                          -hls pixel
float crt[3];                          -crt pixel
float levels;                          -number of digital count levels

int Lab_LabLCh(Lab, LabLCh)
float Lab[3];                          -CIELAB pixel
float LabLCh[3];                       -CIELAB L*C*h°  pixel

int Lab_XYZ(Lab, XYZ, white_point)
float Lab[3];                          -CIELAB pixel
float XYZ[3];                          -XYZ pixel
float white_point[3];                  -white point values

int LabLCh_Lab(LabLCh, Lab)
float LabLCh[3];                       -CIELAB L*C*h°  pixel
float Lab[3];                          -CIELAB pixel

int Luv_LuvLCh(Luv, LuvLCh)
float Luv[3];                          -CIELUV pixel
float LuvLCh[3];                       -CIELUV L*C*h°  pixel

int Luv_XYZ(Luv, XYZ, Yn, unprime, vnprime)
float Luv[3];                          -CIELUV pixel
float XYZ[3];                          -XYZ pixel
float Yn, unprime, vnprime;            -white point values

int LuvLCh_Luv(LuvLCh, Luv)
float LuvLCh[3];                       -CIELUV L*C*h°  pixel
float Luv[3];                          -CIELUV pixel

int scn_XYZ(scn, XYZ, levels, scnmodel)
int scn[3];                            -scanner pixel
float XYZ[3];                          -XYZ pixel
float levels;                          -number of digital count levels
shaper_ptr scnmodel;                   -scanner model structure

int XYZ_crt(XYZ, crt, levels, crtmodel, crt3x3)
float XYZ[3];                          -XYZ pixel
int crt[3];                            -CRT pixel
float levels;                          -number of digital count levels
shaper_ptr crtmodel;                   -crt model structure
shaper_ptr crt3x3;                     -crt 3x3 structure
```

```
int XYZ_Lab(XYZ, Lab, white_point)
float XYZ[3];                    -XYZ pixel
float Lab[3];                    -CIELAB pixel
float white_point[3];            -white point values

int XYZ_Luv(XYZ, Luv, Yn, unprime, vnprime)
float XYZ[3];                    -XYZ pixel
float Luv[3];                    -CIELUV pixel
float Yn, unprime, vnprime;      -white point values

int XYZ_YIQ(XYZ, YIQ, levels, YIQmodel, YIQinv3x3)
float XYZ[3];                    -XYZ pixel
float YIQ[3];                    -YIQ pixel
float levels;                    -number of digital count levels
shaper_ptr YIQmodel;             -YIQ model structure
shaper_ptr YIQinv3x3;            -YIQ inverse 3x3 structure

int XYZ_Yxy(XYZ, Yxy)
float XYZ[3];                    -XYZ pixel
float Yxy[3];                    -Yxy pixel

int YIQ_XYZ(YIQ, XYZ, levels, YIQmodel, YIQ3x3)
float XYZ[3];                    -XYZ pixel
float YIQ[3];                    -YIQ pixel
float levels;                    -number of digital count levels
shaper_ptr YIQmodel;             -YIQ model structure
shaper_ptr YIQinv3x3;            -YIQ 3x3 structure

int Yxy_XYZ(Yxy, XYZ)
float Yxy[3];                    -Yxy pixel
float XYZ[3];                    -XYZ pixel
```

## DESCRIPTION

Normally, the crs_cst routine is called to convert an entire image from one color space to another. The XYZ tristimulus space is used as a common denominator the conversions. To convert from device spaces, the appropriate models must be loaded.

## FILES

/usr/local/lib/libcrs.a

## NAME

libccf -crs routines to perform color manipulations

## DESCRIPTION

This library provides basic transfer function support. Routines beginning with load_ load the necessary parameters for the function and routines beginning with exec_ execuate the function on a pixel. Normally a function is loaded for each dimension of the color space and exec_shps is called to execute the functions on the entire crs image.

## FUNCTION LIST

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| exec_3x1 | exec_function(3) | -execute a 3x1 matrix on a crs pixel |
| exec_3x3 | exec_function(3) | -execute a 3x3 matrix on a crs pixel |
| exec_clip | exec_function(3) | -execute a clipping funciton on a crs pixel |
| exec_crtm | exec_function(3) | -execute a CRT gamma-offset-gain model on a crs pixel |
| exec_imfact | exec_function(3) | -execute an inverse multiplicative factor function |
| exec_mfact | exec_function(3) | -execute a multiplicative factor function on a crs pixel |
| exec_pimf | exec_function(3) | -execute a power/inverse multiplicative factor funciton |
| exec_power | exec_function(3) | -execute a power function on a crs pixel |
| exec_quant | exec_function(3) | -execute a quantizer function on a crs pixel |
| exec_scrv | exec_function(3) | -execute an sigmoidal function on a crs pixel |
| exec_set | exec_function(3) | -execute a constant function on a crs pixel |
| exec_shps | exec_function(3) | -execute a set of transfer functions on a crs image |
| exec_stat | exec_function(3) | -execute a simpel statistical functions on a crs pixel |
| exec_voff | exec_function(3) | -execute an additive offset function on a crs pixel |
| exec_wrap | exec_function(3) | -execute a wraparound function on a crs pixel |
| init_shps | init_function(3) | -initialize a set of transfer functions |
| load_3x1 | load_function(3) | -load a 3x1 matrix function |
| load_3x3 | load_function(3) | -load a 3x3 matrix function |
| load_cal | load_function(3) | -load CRT gamma-offset-gain calibration data |
| load_clip | load_function(3) | -load a clipping function |
| load_crtm | load_function(3) | -load a CRT gamma-offset-gain function |
| load_imfact | load_function(3) | -load an inverse multiplicative factor function |
| load_lmts | load_function(3) | -load crs color space dimensional limits |
| load_mfact | load_function(3) | -load a multiplicative factor function |
| load_null | load_function(3) | -load a null function |
| load_pimf | load_function(3) | -load a power/inverse multiplicative factor function |
| load_power | load_function(3) | -load a power function |
| load_quant | load_function(3) | -load a quantizer function |
| load_scrv | load_function(3) | -load a sigmoidal function |
| load_set | load_function(3) | -load a constant function |
| load_voff | load_function(3) | -load an additive offset function |
| load_wrap | load_function(3) | -load a wrap-around function |

**FILES**

/usr/local/lib/libcrs.a crs runtime library

**SEE ALSO**

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

**exec_3x1** -execute a 3x1 matrix on a crs pixel
**exec_3x3** -execute a 3x3 matrix on a crs pixel
**exec_clip** -execute a clipping funciton on a crs pixel
**exec_crtm** -execute a CRT gamma-offset-gain model on a crs pixel
**exec_imfact** -execute an inverse multiplicative factor function
**exec_mfact** -execute a multiplicative factor function on a crs pixel
**exec_pimf** -execute a power/inverse multiplicative factor funciton
**exec_power** -execute a power function on a crs pixel
**exec_quant** -execute a quantizer function on a crs pixel
**exec_scrv** -execute an sigmoidal function on a crs pixel
**exec_set** -execute a constant function on a crs pixel
**exec_shps** -execute a set of transfer functions on a crs image
**exec_stat** -execute a simpel statistical functions on a crs pixel
**exec_voff** -execute an additive offset function on a crs pixel
**exec_wrap** -execute a wraparound function on a crs pixel

## SYNOPSIS

```
#include <crs.h>

int exec_3x1(pixel, function)
float *pixel;
shaper_ptr function;

int exec_3x3(pixel, function)
float *pixel;
shaper_ptr function;

int exec_clip(pixel, function)
float *pixel;
shaper_ptr function;

int exec_crtmodel(pixel, levels, function)
float *pixel;
float levels;
shaper_ptr function;

int exec_imfactor(pixel, function)
float *pixel;
shaper_ptr function;

int exec_mfactor(pixel, function)
float *pixel;
shaper_ptr function;
```

```
int exec_pimf(pixel, function, min, range)
float *pixel;
shaper_ptr function;
float min;
float range;

int exec_power(pixel, function, min, range)
float *pixel;
shaper_ptr function;
float min;
float range;

int exec_quant(pixel, function)
float *pixel;
shaper_ptr function;

int exec_scurve(pixel, function, min, range)
float *pixel;
shaper_ptr function;
float min;
float range;

int exec_set(pixel, function)
float *pixel;
shaper_ptr function;

int exec_shapers(crs, win, function_set)
crsimg_ptr crs;
actwin_ptr, win;
shaperset_ptr function_set;

int exec_voffset(pixel, function)
float *pixel;
shaper_ptr function;

int exec_wrap(pixel, function)
float *pixel;
shaper_ptr function;

int exec_xmfactor(pixel, function)
float *pixel;
shaper_ptr function;
```

DESCRIPTION

The function exec_shapers executes a set of three transfer functions on an entire crs image. The other functions in this library are called by exec_shapers to execute a single function on a single pixel in a single color space dimension.

FILES

/usr/local/lib/libcrs.a

## NAME

**init_shapers** -initialize a set of transfer functions

## SYNOPSIS

```
#include <crs.h>

int init_shapers(function_set)
shaperset_ptr function_set;
```

## DESCRIPTION

This routine initializes a set of three transfer functions to the NULL function.

## FILES

/usr/local/lib/libcrs.a

## NAME

| | |
|---|---|
| **load_3x1** | -load a 3x1 matrix function |
| **load_3x3** | -load a 3x3 matrix function |
| **load_cal** | -load CRT gamma-offset-gain calibration data |
| **load_clip** | -load a clipping function |
| **load_crtm** | -load a CRT gamma-offset-gain function |
| **load_imfact** | -load an inverse multiplicative factor function |
| **load_lmts** | -load crs color space dimensional limits |
| **load_mfact** | -load a multiplicative factor function |
| **load_null** | -load a null function |
| **load_pimf** | -load a power/inverse multiplicative factor function |
| **load_power** | -load a power function |
| **load_quant** | -load a quantizer function |
| **load_scrv** | -load a sigmoidal function |
| **load_set** | -load a constant function |
| **load_voff** | -load an additive offset function |
| **load_wrap** | -load a wrap-around function |

## SYNOPSIS

```
#include <crs.h>

int load_3x1(function, ch1, ch2, ch3)
shaper_ptr function;
float ch1, ch2, ch3;

int load_3x3(function, xy00, xy01, xy02, xy10, xy11, xy12, xy20, xy21, xy22)
shaper_ptr function;
float xy00, xy01, xy02, xy10, xy11, xy12, xy20, xy21, xy22;

int load_clip(function, min, max)
shaper_ptr function;
float min, max;

int load_crtcal(filename, crtmodel, from3x3, to3x3)
char filename[];
shaper_ptr crtmodel;
shaper_ptr from3x3;
shaper_ptr to3x3;

int load_crtmodel(function, red_offset, red_gain, red_gamma, green_offset,
                  green_gain, green_gamma, blue_offset, blue_gain, blue_gamma)
shaper_ptr function;
float red_offset, red_gain, red_gamma;
float green_offset, rgreen_gain, green_gamma;
float blue_offset, blue_gain, blue_gamma;

int load_imfactor(function, mfactor, offset)
```

```
shaper_ptr function;
float mfactor, offset;

int load_mfactor(function, mfactor)
shaper_ptr function;
float mfactor;

int load_null(function)
shaper_ptr function;

int load_pimf(function, power, mfactor, offset)
shaper_ptr function;
float power, mfactor, offset;

int load_power(function, power)
shaper_ptr function;
float power;

int load_quant(function, min, max, num_bins)
shaper_ptr function;
float min, max;
int num_bins;

int load_scurve(function, sfactor)
shaper_ptr function;
float sfactor;

int load_set(function, factor)
shaper_ptr function;
float factor;

int load_voffset(function, offset)
shaper_ptr function;
float offset;

int load_wrap(function, min, max)
shaper_ptr function;
float min, max;

int load_xmfactor(function, mfactor, offset)
shaper_ptr function;
float mfactor, offset;
```

DESCRIPTION

These routines load the necessary parameters for the transfer functions.

FILES

/usr/local/lib/libcrs.a

## NAME

libfio -crs routines to perform file input/output

## DESCRIPTION

This library provides disk input/output support for various image storage formats.

## FUNCTION LIST

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| load_crs | load_crs(3) | -load a crs image from disk |
| load_crsh | load_crs(3) | -load a crs image header |
| load_crsi | load_crs(3) | -load a crs image data |
| load_mds | load_crs(3) | -load an mds image from disk |
| load_ppm | load_crs(3) | -load a ppm image from disk |
| load_rgb | load_crs(3) | -load a raw rgb image from disk |
| load_xwd | load_crs(3) | -load an X11 image from disk |
| save_crs | save_crs(3) | -save a crs image to disk |
| save_crsh | save_crs(3) | -save a crs image header |
| save_crsi | save_crs(3) | -save crs image data |
| save_mds | save_crs(3) | -save an mds image to disk |
| save_pix | save_crs(3) | -save a pixar image to disk |
| save_ppm | save_crs(3) | -save a ppm image to disk |
| save_xwd | save_crs(3) | -save an X11 image to disk |

## FILES

/usr/local/lib/libcrs.a crs runtime library

## SEE ALSO

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

| | |
|---|---|
| **load_crs** | -load a crs image from disk |
| **load_crsh** | -load a crs image header |
| **load_crsi** | -load a crs image data |
| **load_mds** | -load an mds image from disk |
| **load_pix** | -load a raw pixar image from disk |
| **load_ppm** | -load a ppm image from disk |
| **load_rgb** | -load a raw rgb image from disk |
| **load_xwd** | -load an X11 image from disk |

## SYNOPSIS

```
#include <crs.h>

int load_crs(filename, crs)
char *filename;
crsimg_ptr crs;

int load_crshdr(ifn, crs)
int ifn;
crsimg_ptr crs;

int load_crsimg(ifn, crs)
int ifn;
crsimg_ptr crs;

int load_crt(filename, crt)
char *filename;
crsimg_ptr crs;

int load_mds(filename, mds)
char *filename;
mdsimg_ptr mds;

int load_pix(filename, pix)
char *filename;
piximg_ptr pix;

int load_ppm(filename, ppm)
char *filename;
ppmimg_ptr ppm;

int load_rgb(filename, ppm)
char *filename;
ppmimg_ptr ppm;

int load_xwd(filename, xwd)
char *filename;
xwdimg_ptr xwd;
```

## DESCRIPTION

These routines load a stored disk image into the appropriate image structure in memory.

## FILES

/usr/local/lib/libcrs.a

## NAME

| | |
|---|---|
| **save_crs** | -save a crs image to disk |
| **save_crsh** | -save a crs image header |
| **save_crsi** | -save crs image data |
| **save_mds** | -save an mds image to disk |
| **save_pix** | -save a pixar image to disk |
| **save_ppm** | -save a ppm image to disk |
| **save_xwd** | -save an X11 image to disk |

## SYNOPSIS

```
#include <crs.h>

int save_crs(filename, crs)
char *filename;
crsimg_ptr crs;

int save_crshdr(ifn, crs)
int ifn;
crsimg_ptr crs;

int save_crsimg(ifn, crs)
int ifn;
crsimg_ptr crs;

int save_crt(filename, crt)
char *filename;
crsimg_ptr crs;

int save_mds(filename, mds)
char *filename;
mdsimg_ptr mds;

int save_pix(filename, pix)
char *filename;
piximg_ptr pix;

int save_ppm(filename, ppm)
char *filename;
ppmimg_ptr ppm;

int save_xwd(filename, xwd)
char *filename;
xwdimg_ptr xwd;
```

## DESCRIPTION

These routines load a stored disk image into the appropriate image structure in

memory.

**FILES**

/usr/local/lib/libcrs.a

**NAME**

libimf -crs routines to perform image manipulations

**DESCRIPTION**

These functions provide basic image manipulation routines for crs images.

**FUNCTION LIST**

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| crs_clear | crs_clear(3) | -clear a crs image and header |
| crs_crop | crs_clear(3) | -crop a crs image |
| crs_flipx | crs_clear(3) | -flip crs image horizontally |
| crs_move | crs_clear(3) | -move crs image within active window |
| crs_scale | crs_clear(3) | -scale crs image within active window |

**FILES**

/usr/local/lib/libcrs.a crs runtime library

**SEE ALSO**

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

| | |
|---|---|
| **crs_clear** | -clear a crs image and header |
| **crs_crop** | -crop a crs image |
| **crs_flipx** | -flip crs image horizontally |
| **crs_move** | -move crs image within active window |
| **crs_scale** | -scale crs image within active window |

## SYNOPSIS

```
#include <crs.h>

int crs_move(crs, atx, aty)
crsimg_ptr crs;
int atx, aty;

int crs_crop(crs, cropped, win)
crsimg_ptr crs, cropped;
actwin_ptr win;

int crs_scale(crs, scaled, win)
crsimg_ptr crs, scaled;
actwin_ptr win;

int crs_flipx(crs, win)
crsimg_ptr crs;
actwin_ptr win;

int crs_clear(crs, color)
crsimg_ptr crs;
float color[3];
```

## DESCRIPTION

These functions provide basic image manipulation routines for crs images.

## FILES

/usr/local/lib/libcrs.a

## NAME

libcra -crs routines to perform color reproduction analysis

## DESCRIPTION

This library provides basic image analysis routines for crs images using tools functions that are internally identical to the transfer function concept. Normally, a tool is initialized, exec_tools is called to execute a set of three tools on the color space dimensions, and finally the calcuations are made.

## FUNCTION LIST

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| calc_dCMC | calc_tool(3) | -calculate CMC color differences for a pair of pixels |
| calc_dE | calc_tool(3) | -calculate CIELAB color differences for a pair of pixels |
| calc_dMCSL | calc_tool(3) | -calculate MCSL color differences for a pair of pixels |
| calc_dMCSL2 | calc_tool(3) | -calculate MCSL2 color differences for a pair of pixels |
| calc_stat | calc_tool(3) | -calculate simple statistics for a pixel |
| exec_dCMC | exec_tool(3) | -execute a CMC color difference function on a crs pixel |
| exec_dE | exec_tool(3) | -execute a CIELAB color difference function on a crs pixel |
| exec_dMCSL | exec_tool(3) | -execute an MCSL color difference function on a crs pixel |
| exec_dMCSL2 | exec_tool(3) | -execute an MCSL2 color difference function on a crs pixel |
| exec_hist | exec_tool(3) | -execute a histogram function on a crs pixel |
| exec_tools | exec_tool(3) | -execute an analytical tool on a crs image |
| init_dCMC | init_tool(3) | -initialize a CMC color difference function |
| init_dE | init_tool(3) | -initialize a CIELAB color difference function |
| init_dMCSL | init_tool(3) | -initialize an MCSL color difference function |
| init_dMCSL2 | init_tool(3) | -initialize an MCSL2 color difference function |
| init_hist | init_tool(3) | -initialize a histogram function |
| init_stat | init_tool(3) | -initialize a simple statistical function |

## FILES

/usr/local/lib/libcrs.a crs runtime library

## SEE ALSO

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

**calc_dCMC** -calculate CMC color differences for a pair of pixels

**calc_dE** -calculate CIELAB color differences for a pair of pixels

**calc_dMCSL** -calculate MCSL color differences for a pair of pixels

**calc_dMCSL2** -calculate MCSL2 color differences for a pair of pixels

**calc_stat** -calculate simple statistics for a pixel

**calc_hist** -calculate histogram for a pixel

## SYNOPSIS

```
#include <crs.h>

int calc_dCMC(tool)
shaper_ptr tool;

int calc_dE(tool)
shaper_ptr tool;

int calc_dMCSL(tool)
shaper_ptr tool;

int calc_dMCSL2(tool)
shaper_ptr tool;

int calc_stat(tool)
shaper_ptr tool;
```

## DESCRIPTION

The routines perform the final calculations necessary for the appropriate image analysis.

## FILES

/usr/local/lib/libcrs.a

## NAME

**exec_dCMC** -execute a CMC color difference function on a crs pixel

**exec_dE** -execute a CIELAB color difference function on a crs pixel

**exec_dMCSL** -execute an MCSL color difference function on a crs pixel

**exec_dMCSL2** -execute an MCSL2 color difference function on a crs pixel

**exec_hist** -execute a histogram function on a crs pixel

**exec_tools** -execute a set of analytical tools on a crs image

## SYNOPSIS

```
#include <crs.h>

int exec_dCMC(orig, dupe, tool)
float orig[3], dupe[3];
shaper_ptr tool;

int exec_dE(orig, dupe, tool)
float orig[3], dupe[3];
shaper_ptr tool;

int exec_dMCSL(orig, dupe, tool)
float orig[3], dupe[3];
shaper_ptr tool;

int exec_dMCSL2(orig, dupe, tool)
float orig[3], dupe[3];
shaper_ptr tool;

int exec_hist(pixel, tool)
float pixel[3];
shaper_ptr tool;

int exec_stat(pixel, tool)
float pixel[3];
shaper_ptr tool;

int exec_tools(orig, dupe, win, tool_set)
crsimg_ptr orig, dupe;
actwin_ptr win;
shaperset_ptr tool_set;
```

## DESCRIPTION

The routine exec_tools executes a set of three tools on a crs image. The other functions are called by exec_tools in order to operate on a single pixel in a single color space dimension at

a time.

**FILES**

/usr/local/lib/libcrs.a

## NAME

| | |
|---|---|
| **init_dCMC** | -initialize a CMC color difference function |
| **init_dE** | -initialize a CIELAB color difference function |
| **init_dMCSL** | -initialize an MCSL color difference function |
| **init_dMCSL2** | -initialize an MCSL2 color difference function |
| **init_hist** | -initialize a histogram function |
| **init_stat** | -initialize a simple statistical function |

## SYNOPSIS

```
#include <crs.h>

int init_dCMC(tool, l_value, c_value)
shaper_ptr tool;
float l_value, c_value;

int init_dE(tool)
shaper_ptr tool;

int init_dMCSL(tool)
shaper_ptr tool;

int init_dMCSL2(tool)
shaper_ptr tool;

int init_hist(tool, min_1, range_1, num_bins_1,
              min_2, range_2, num_bins_2, min_3, range_3, num_bins_3)
shaper_ptr tool;
float min_1, range_1, num_bins_1;
float min_2, range_2 num_bins_2;
float min_3, range_3, num_bins_3;

int init_stat(tool)
shaper_ptr tool;
```

## DESCRIPTION

These functions initialize the tools to a set state ready for execution.

## FILES

/usr/local/lib/libcrs.a

## NAME

libdcf -crs routines to perform device characterization functions

## DESCRIPTION

This library provides device dependent support routines to convert images between device dependent color spaces and device independent color spaces.

## FUNCTION LIST

| *Name* | *Appears on Page* | *Description* |
|---|---|---|
| howtek2crs | howtek2crs(3) | -scan a crs image from howtek scanner |
| howtek2ppm | howtek2crs(3) | -scan a ppm image from howtek scanner |
| crs2hp | crs2hp(3) | -display crs image onto an hp starbase display |
| mds2hp | crs2hp(3) | -display mds image onto an hp starbase display |
| ppm2hp | crs2hp(3) | -display ppm image onto an hp starbase display |
| hp2crs | crs2hp(3) | -convert an hp starbase display into a crs image |
| hp2mds | crs2hp(3) | -convert an hp starbase display into an mds image |
| hp2ppm | crs2hp(3) | -convert an hp starbase display into a ppm image |
| crs2matrix | crs2matrix(3) | -expose a crs image onto matrix film recorder |
| ppm2matrix | crs2matrix(3) | -expose a ppm image onto matrix film recorder |
| crs2taac | crs2taac(3) | -display crs image onto a sun TAAC display |
| mds2taac | crs2taac(3) | -display mds image onto a sun TAAC display |
| ppm2taac | crs2taac(3) | -display ppm image onto a sun TAAC display |
| taac2crs | crs2taac(3) | -convert a sun TAAC image into a crs image |
| taac2mds | crs2taac(3) | -convert a sun TAAC image into an mds image |
| taac2ppm | crs2taac(3) | -convert a sun TAAC image into a ppm image |

## FILES

/usr/local/lib/libcrs.a crs runtime library

## SEE ALSO

libmisc(3), libcst(3), libccf(3), libfio(3), libimf(3), libcra(3), libdcf(3)

## NAME

**howtek2crs** -scan a crs image from howtek scanner
**howtek2ppm** -scan a ppm image from howtek scanner

## SYNOPSIS

```
#include <crs.h>

int howtek2crs(crs, win, howtekmodel)
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr howtekmodel;

int howtek2ppm(ppm, win,)
ppmimg_ptr ppm;
actwin_ptr win;
```

## DESCRIPTION

These functions scan an image scanned from a Howtek scanner into either a raw ppm image or a device independent crs image.

## FILES

/usr/local/lib/libcrs.a

## NAME

**crs2hp** -display crs image onto an hp starbase display
**mds2hp** -display mds image onto an hp starbase display
**ppm2hp** -display ppm image onto an hp starbase display
**hp2crs** -convert an hp starbase display into a crs image
**hp2mds** -convert an hp starbase display into an mds image
**hp2ppm** -convert an hp starbase display into a ppm image

## SYNOPSIS

```
#include <crs.h>

int crs2hp(crs, win, crtmodel, crt3x3)
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int mds2hp(mds, win, crtmodel, crt3x3)
mdsimg_ptr mds;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int ppm2hp(ppm, win, crtmodel, crt3x3)
ppmimg_ptr ppm;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int hp2crs(crs, win, crtmodel, crt3x3)
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int hp2mds(mds, win, crtmodel, crt3x3)
mdsimg_ptr mds;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int hp2ppm(ppm, win, crtmodel, crt3x3)
ppmimg_ptr ppm;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;
```

**DESCRIPTION**

These functions convert between various image formats and an HP starbase display.

**FILES**

/usr/local/lib/libcrs.a

## NAME

**crs2matrix** -expose a crs image onto matrix film recorder
**ppm2matrix** -expose a ppm image onto matrix film recorder

## SYNOPSIS

```
#include <crs.h>

int crs2matrix(crs, win, matrixmodel)
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr matixmodel;

int ppm2matrix(ppm, win)
ppmimg_ptr ppm;
actwin_ptr win;
```

## DESCRIPTION

These functions expose an image on a Matrix film recorder from either a raw ppm image or a device independent crs image.

## FILES

/usr/local/lib/libcrs.a

## NAME

| | |
|---|---|
| **crs2taac** | -display crs image onto a sun TAAC display |
| **mds2taac** | -display mds image onto a sun TAAC display |
| **ppm2taac** | -display ppm image onto a sun TAAC display |
| **taac2crs** | -convert a sun TAAC image into a crs image |
| **taac2mds** | -convert a sun TAAC image into an mds image |
| **taac2ppm** | -convert a sun TAAC image into a ppm image |

## SYNOPSIS

```
#include <crs.h>

int crs2taac(crs, win, crtmodel, crt3x3)
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int mds2taac(mds, win, crtmodel, crt3x3)
mdsimg_ptr mds;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int ppm2taac(ppm, win, crtmodel, crt3x3)
ppmimg_ptr ppm;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int taac2crs(crs, win, crtmodel, crt3x3)
crsimg_ptr crs;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int taac2mds(mds, win, crtmodel, crt3x3)
mdsimg_ptr mds;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;

int taac2ppm(ppm, win, crtmodel, crt3x3)
ppmimg_ptr ppm;
actwin_ptr win;
shaper_ptr crtmodel;
shaper_ptr crt3x3;
```

## DESCRIPTION

These functions convert between various image formats and an Sun TAAC display.

**FILES**

/usr/local/lib/libcrs.a

# Appendix H : CRS Utility Programs

| Program | Description |
|---|---|
| cmf | execute a chroma multiplicative factor on a crs image |
| cpower | execute a chroma power function on a crs image |
| cq | execute a chroma quantizer function on a crs image |
| hlsgain | execute a chroma gain function on an hls image |
| hoh | execute a hue additive offset function on a crs image |
| hq | execute a hue quantizer function on a crs image |
| limf | execute a lightness multiplicative factor on a crs image |
| lpimf | execute a lightness power function and an inverse multiplicative factor on a crs image |
| lpower | execute a lightness power function on a crs image |
| lq | execute a lightness quantizer function on a crs image |
| lscrv | execute a lightness sigmoidal function on a crs image |
| rgbgain | execute a chroma gain function on an rgb image |
| setlstar | execute a lightness constant function on a crs image |
| yiqgain | execute a chroma gain function on an YIQ image |
| chgspace | change a crs image from one color space to another |
| chgformat | change one image format to another |
| makefinalpix | create experimental images from a set of images and parameter files |
| runflicker | run sequential paired comparison experiment |
| ppm2hp | display a ppm image onto an HP display |
| calcdE | calculate various color difference metrics on a crs image |
| hp2ppm | capture a ppm image from an HP display |

# Appendix I : C Shell Utility Examples

## Shell script to set-up experimental.

```
#!/bin/csh

randlines imglist.h imglist.h2
randlines imglist.h2 imglist.hrnd
randlines imglist.v imglist.v2
randlines imglist.v2 imglist.vrnd
mergelines imglist $1
rm imglist.?rnd
```

## Shell script to run experiment.

```
#!/bin/csh

video -sony
chmap -i
gamma 1
date >$1
runflicker imglist.$1 $2
date >> $1
```

## Shell Script to create demonstration images (in tiff for transportation purposes).

```
chgformat MikeFruit.ppm crs MikeFruitrs

lscrv  MikeFruitrs 1.20      slide01rs
lpower MikeFruitrs 1.20      slide02rs
lpimf  MikeFruitrs 1.20 0.80 slide03rs
lq     MikeFruitrs 256       slide04rs
lq     MikeFruitrs 128       slide05rs
lq     MikeFruitrs 64        slide06rs
lq     MikeFruitrs 32        slide07rs
lq     MikeFruitrs 16        slide08rs

cmf    MikeFruitrs 0.80      slide09rs
cpower MikeFruitrs 0.80      slide10rs
cq     MikeFruitrs 256       slide11rs
cq     MikeFruitrs 128       slide12rs
cq     MikeFruitrs 64        slide13rs
cq     MikeFruitrs 32        slide14rs
```

```
cq    MikeFruitrs 16      slide15rs
cq    MikeFruitrs 8       slide16rs
cq    MikeFruitrs 4       slide17rs


lxmf  MikeFruitrs 0.75    slide18rs
cmf   slide18rs  0.75     slide19rs
hoh   slide19rs  20.0     slide20rs
hoh   slide19rs  -20.0    slide21rs
hq    MikeFruitrs 256     slide22rs
hq    MikeFruitrs 64      slide23rs
hq    MikeFruitrs 16      slide24rs
hq    MikeFruitrs 8       slide25rs
hq    MikeFruitrs 4       slide26rs
hq    MikeFruitrs 2       slide27rs

chgformat slide01rs ppm slide01.ppm
chgformat slide02rs ppm slide02.ppm
chgformat slide03rs ppm slide03.ppm
chgformat slide04rs ppm slide04.ppm
chgformat slide05rs ppm slide05.ppm
chgformat slide06rs ppm slide06.ppm
chgformat slide07rs ppm slide07.ppm
chgformat slide08rs ppm slide08.ppm
chgformat slide09rs ppm slide09.ppm
chgformat slide10rs ppm slide10.ppm
chgformat slide11rs ppm slide11.ppm
chgformat slide12rs ppm slide12.ppm
chgformat slide13rs ppm slide13.ppm
chgformat slide14rs ppm slide14.ppm
chgformat slide15rs ppm slide15.ppm
chgformat slide16rs ppm slide16.ppm
chgformat slide17rs ppm slide17.ppm
chgformat slide18rs ppm slide18.ppm
chgformat slide19rs ppm slide19.ppm
chgformat slide20rs ppm slide20.ppm
chgformat slide21rs ppm slide21.ppm
chgformat slide22rs ppm slide22.ppm
chgformat slide23rs ppm slide23.ppm
chgformat slide24rs ppm slide24.ppm
chgformat slide25rs ppm slide25.ppm
chgformat slide26rs ppm slide26.ppm
chgformat slide27rs ppm slide27.ppm

pnmtotiff slide01.ppm > slide01.tiff
pnmtotiff slide02.ppm > slide02.tiff
```

```
pnmtotiff slide03.ppm > slide03.tiff
pnmtotiff slide04.ppm > slide04.tiff
pnmtotiff slide05.ppm > slide05.tiff
pnmtotiff slide06.ppm > slide06.tiff
pnmtotiff slide07.ppm > slide07.tiff
pnmtotiff slide08.ppm > slide08.tiff
pnmtotiff slide09.ppm > slide09.tiff
pnmtotiff slide10.ppm > slide10.tiff
pnmtotiff slide11.ppm > slide11.tiff
pnmtotiff slide12.ppm > slide12.tiff
pnmtotiff slide13.ppm > slide13.tiff
pnmtotiff slide14.ppm > slide14.tiff
pnmtotiff slide15.ppm > slide15.tiff
pnmtotiff slide16.ppm > slide16.tiff
pnmtotiff slide17.ppm > slide17.tiff
pnmtotiff slide18.ppm > slide18.tiff
pnmtotiff slide19.ppm > slide19.tiff
pnmtotiff slide20.ppm > slide20.tiff
pnmtotiff slide21.ppm > slide21.tiff
pnmtotiff slide22.ppm > slide22.tiff
pnmtotiff slide23.ppm > slide23.tiff
pnmtotiff slide24.ppm > slide24.tiff
pnmtotiff slide25.ppm > slide25.tiff
pnmtotiff slide26.ppm > slide26.tiff
pnmtotiff slide27.ppm > slide27.tiff
```

# Appendix J : R. R. Donnelley Experimental Results

A repetition of the experiment was performed at R. R. Donnelley in Lisle, Illinois. The same environment, experimental procedures, and images were used in both the RIT experiment and the Donnelley experiment. The primary differences were the number of observers and observer experience. The number of observers in the Donnelley experiment was one-third the number in the RIT experiment. More importantly, the observers' experience in judging color images in the Donnelley experiment ranged from two to over twenty years. The results from this experiment coincide well with the RIT experiment. The exceptions are due to observer experience.

Table J.1 shows the dramatic decrease in observer noise in the Donnelley experiment. The best observer (MCE) was an experienced master color printer. Tables J.2 and J.3 show the perceptibility and acceptability results. Overall these results correlate well with the RIT results except the Donnelley observers are more sensitive to changes in color. This is dramatically shown in the perceptibility scene results in Table J.3 and Graphs J.1 through J.10. The lack of fiducial limits and complete lack of statistical analysis in the case of hue angle for scene #1 are due to the overwhelming majority of the responses indicating a perceptible difference. This could be corrected by adding smaller changes in the parameter levels, but the pilot experiment from which these levels were derived did not include such experienced observers. Again, the tolerances are fairly symmetric allowing for a reduction of transfer functions in future experiments. Overall the Donnelley observers were 4 to 5 percent more sensitive than their less experienced counterparts at RIT.

The most significant result from the Donnelley experiment is

illustrated in both the perceptibility and acceptability tolerances by scene. These results are shown in Graphs J.1 through J.20. The acceptability tolerances by scene shown in Graphs J.11 through J.20 show that experienced observers do not judge color differences by scene content. These results were found in the RIT experiment, but not to the significance shown below. This implies that future experiments can safely be limited to a small number of scenes with little regard to scene content. Additionally, this experiment emphasizes that with experience, observers depend less and less on specific areas of a scene and more and more on the overall scene appearance.

Table J.1 : Observer's Chi-Squared Statistics for Donnelley Experiment

| Observer | Perceptibility | | Acceptability | |
|---|---|---|---|---|
| | Chi $^2$ | Pr > Chi $^2$ | Chi $^2$ | Pr > Chi $^2$ |
| buk | 1.49 | 0.85 | 3.10 | 0.76 |
| dun | 2.22 | 0.76 | 2.51 | 0.76 |
| fli | 4.00 | 0.60 | 4.24 | 0.69 |
| mce | 0.54 | 0.96 | 0.67 | 0.95 |
| non | 3.07 | 0.72 | 1.63 | 0.83 |
| par | 1.99 | 0.78 | 1.26 | 0.90 |
| prk | 1.40 | 0.87 | 2.29 | 0.76 |
| rod | 0.78 | 0.93 | 4.07 | 0.70 |
| shi | 1.25 | 0.88 | 2.74 | 0.73 |
| sta | 1.05 | 0.93 | 2.16 | 0.82 |
| xie | 2.22 | 0.81 | 2.42 | 0.77 |

Table J.2 : Perceptibility Results for Donnelley Experiment

| Function | T50 | LOWER | UPPER | Pr > Chi $^2$ |
|---|---|---|---|---|
| LMF | 0.95 | 0.94 | 0.96 | 0.95 |
| LPH | 1.06 | 1.04 | 1.08 | 0.98 |
| LPL | 0.95 | 0.93 | 0.97 | 0.77 |
| LSH | 1.09 | 1.07 | 1.11 | 0.85 |
| LSL | 0.92 | 0.90 | 0.94 | 0.83 |
| CMF | 0.94 | 0.93 | 0.96 | 0.43 |
| CPH | 1.13 | 1.11 | 1.14 | 0.17 |
| CPL | 0.88 | 0.86 | 0.90 | 0.90 |
| HOH | 3.20 | 2.35 | 3.75 | 0.99 |
| HOL | -2.82 | -3.64 | -1.32 | 0.34 |

Table J.3 : Acceptability Results for Donnelley Experiment

| Function | T50 | LOWER | UPPER | Pr > Chi $^2$ |
|---|---|---|---|---|
| LMF | 0.89 | 0.89 | 0.90 | 0.83 |
| LPH | 1.14 | 1.12 | 1.15 | 0.13 |
| LPL | 0.90 | 0.90 | 0.91 | 0.38 |
| LSH | 1.21 | 1.20 | 1.23 | 0.52 |
| LSL | 0.87 | 0.86 | 0.88 | 0.91 |
| CMF | 0.89 | 0.88 | 0.90 | 0.73 |
| CPH | 1.24 | 1.21 | 1.28 | 0.04 |
| CPL | 0.83 | 0.82 | 0.84 | 0.15 |
| HOH | 5.25 | 4.80 | 5.70 | 0.52 |
| HOL | -5.67 | -6.24 | -5.06 | 0.95 |

Table J.4 : Perceptibility Results by Scene for Donnelley Experiment

| Scene | # | Function | T50 | LOWER | UPPER | Prob > Chi $^2$ |
|---|---|---|---|---|---|---|
| gcmpnl | 1 | LMF | 0.92 | 0.89 | 0.97 | 0.29 |
| mdfnfh | 2 | LMF | 0.94 | . | . | 0.99 |
| q60mfh | 3 | LMF | 0.95 | 0.91 | 9.15 | 0.62 |
| q60pfh | 4 | LMF | 0.94 | 0.91 | 0.97 | 0.45 |
| rcimnl | 5 | LMF | 0.97 | 0.95 | 3.15 | 0.94 |
| smannl | 6 | LMF | 0.96 | 0.94 | 1.10 | 0.60 |
| gcmpnl | 1 | LPH | 1.08 | . | . | 0.99 |
| mdfnfh | 2 | LPH | 1.06 | . | . | 0.84 |
| q60mfh | 3 | LPH | 1.05 | . | . | 1.00 |
| q60pfh | 4 | LPH | 1.05 | . | . | 1.00 |
| rcimnl | 5 | LPH | 1.05 | . | . | 1.00 |
| smannl | 6 | LPH | 1.09 | . | . | 1.00 |
| gcmpnl | 1 | LPL | 0.95 | . | . | 0.57 |
| mdfnfh | 2 | LPL | 0.94 | . | . | 0.93 |
| q60mfh | 3 | LPL | 0.94 | . | . | 0.93 |
| q60pfh | 4 | LPL | 0.95 | . | . | 0.95 |
| rcimnl | 5 | LPL | 0.95 | . | . | 1.00 |
| smannl | 6 | LPL | 0.96 | . | . | 1.00 |
| gcmpnl | 1 | LSH | 1.07 | 0.82 | 1.11 | 0.96 |
| mdfnfh | 2 | LSH | 1.11 | 0.95 | 1.15 | 0.98 |
| q60mfh | 3 | LSH | 1.09 | 0.92 | 1.13 | 0.95 |
| q60pfh | 4 | LSH | 1.09 | . | . | 1.00 |
| rcimnl | 5 | LSH | 1.03 | 0.09 | 1.14 | 0.70 |
| smannl | 6 | LSH | 1.09 | 0.96 | 1.14 | 0.57 |
| gcmpnl | 1 | LSL | 0.93 | . | . | 1.00 |
| mdfnfh | 2 | LSL | 0.92 | . | . | 0.99 |
| q60mfh | 3 | LSL | 0.93 | . | . | 0.78 |
| q60pfh | 4 | LSL | 0.99 | . | . | 0.20 |
| rcimnl | 5 | LSL | 0.87 | 0.83 | 1.44 | 0.93 |
| smannl | 6 | LSL | 0.90 | 0.87 | 0.98 | 0.49 |
| gcmpnl | 1 | CMF | 0.93 | . | . | 1.00 |
| mdfnfh | 2 | CMF | 0.93 | . | . | 1.00 |
| q60mfh | 3 | CMF | 0.93 | . | . | 0.91 |
| q60pfh | 4 | CMF | 0.94 | . | . | 0.92 |
| rcimnl | 5 | CMF | 1.04 | . | . | 0.07 |
| smannl | 6 | CMF | 0.95 | . | . | 1.00 |
| gcmpnl | 1 | CPH | 1.11 | 1.01 | 1.15 | 0.55 |
| mdfnfh | 2 | CPH | 1.12 | 1.01 | 1.16 | 0.89 |
| q60mfh | 3 | CPH | 1.15 | 1.08 | 1.21 | 0.33 |

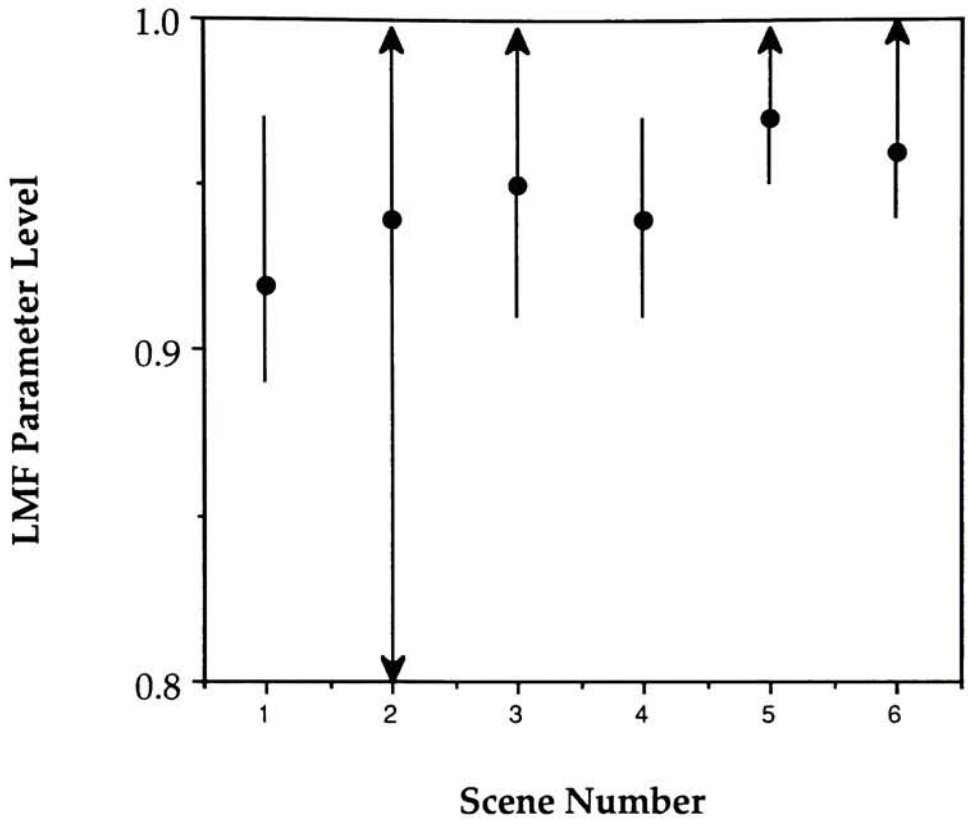
| | | | | | | |
|---|---|---|---|---|---|---|
| q60pfh | 4 | CPH | 1.13 | . | . | 0.96 |
| rcimnl | 5 | CPH | 1.11 | 0.77 | 1.17 | 0.19 |
| smannl | 6 | CPH | 1.19 | 1.12 | 1.25 | 0.33 |
| gcmpnl | 1 | CPL | 0.84 | . | . | 0.05 |
| mdfnfh | 2 | CPL | 0.85 | . | . | 0.82 |
| q60mfh | 3 | CPL | 0.86 | . | . | 0.06 |
| q60pfh | 4 | CPL | 0.91 | 0.86 | 0.98 | 0.71 |
| rcimnl | 5 | CPL | 0.90 | . | . | 0.86 |
| smannl | 6 | CPL | 0.94 | 0.88 | 1.57 | 0.84 |
| gcmpnl | 1 | HOH | | | | |
| mdfnfh | 2 | HOH | 3.37 | . | . | 1.00 |
| q60mfh | 3 | HOH | 3.02 | . | . | 1.00 |
| q60pfh | 4 | HOH | 4.42 | . | . | 0.99 |
| rcimnl | 5 | HOH | 3.15 | . | . | 1.00 |
| smannl | 6 | HOH | 2.32 | . | . | 1.00 |
| gcmpnl | 1 | HOL | | | | |
| mdfnfh | 2 | HOL | -2.38 | . | . | 1.00 |
| q60mfh | 3 | HOL | -4.76 | -7.22 | 3.34 | 0.59 |
| q60pfh | 4 | HOL | -2.18 | . | . | 1.00 |
| rcimnl | 5 | HOL | -3.73 | . | . | 1.00 |
| smannl | 6 | HOL | -2.54 | . | . | 1.00 |

Table J.5 : Acceptability Results by Scene for Donnelley Experiment

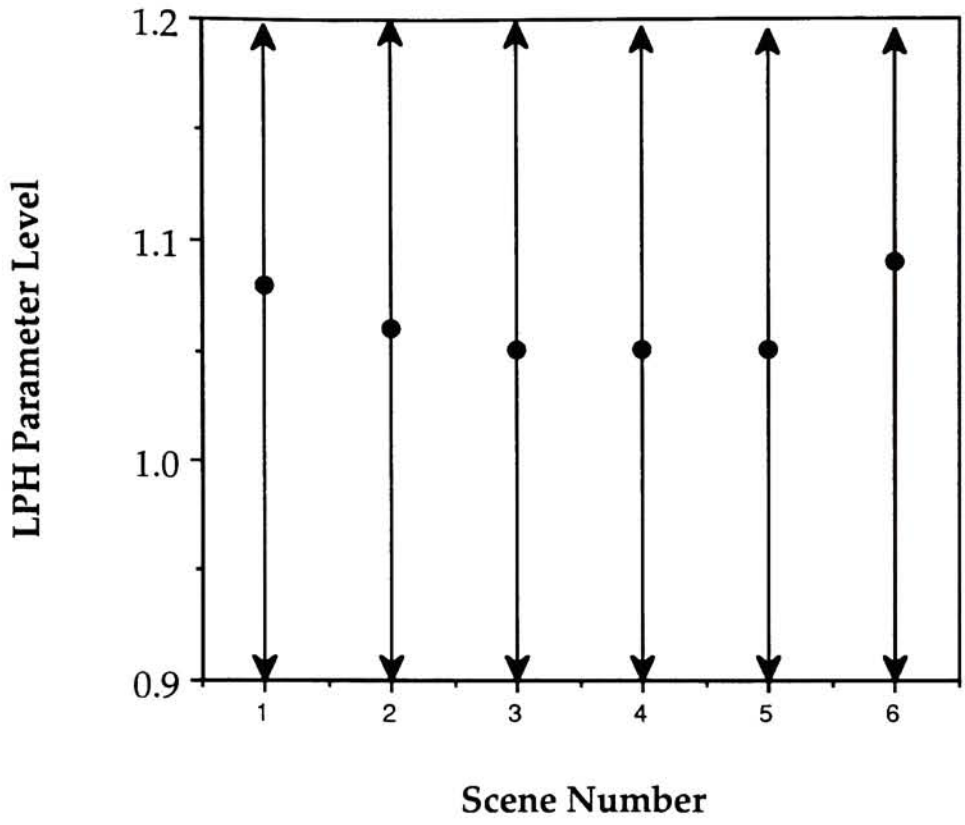
| Scene | # | Function | T50 | LOWER | UPPER | Prob > Chi $^2$ |
|---|---|---|---|---|---|---|
| gcmpnl | 1 | LMF | 0.88 | 0.86 | 0.90 | 0.36 |
| mdfnfh | 2 | LMF | 0.90 | 0.88 | 0.91 | 0.72 |
| q60mfh | 3 | LMF | 0.87 | 0.84 | 0.89 | 0.25 |
| q60pfh | 4 | LMF | 0.90 | 0.88 | 0.91 | 0.53 |
| rcimnl | 5 | LMF | 0.93 | 0.92 | 0.95 | 0.12 |
| smannl | 6 | LMF | 0.88 | 0.85 | 0.90 | 0.65 |
| gcmpnl | 1 | LPH | 1.15 | 1.10 | 1.19 | 0.42 |
| mdfnfh | 2 | LPH | 1.11 | 1.07 | 1.14 | 0.99 |
| q60mfh | 3 | LPH | 1.14 | 1.09 | 1.18 | 0.53 |
| q60pfh | 4 | LPH | 1.13 | 1.09 | 1.16 | 0.67 |
| rcimnl | 5 | LPH | 1.14 | 1.07 | 1.19 | 0.66 |
| smannl | 6 | LPH | 1.17 | 1.12 | 1.20 | 0.92 |
| gcmpnl | 1 | LPL | 0.91 | 0.90 | 0.93 | 1.00 |
| mdfnfh | 2 | LPL | 0.90 | 0.88 | 0.92 | 1.00 |
| q60mfh | 3 | LPL | 0.90 | 0.88 | 0.91 | 0.99 |
| q60pfh | 4 | LPL | 0.91 | 0.90 | 0.94 | 0.38 |

| | | | | | | |
|---|---|---|---|---|---|---|
| rcimnl | 5 | LPL | 0.90 | 0.88 | 0.91 | 0.99 |
| smannl | 6 | LPL | 0.90 | 0.88 | 0.92 | 0.76 |
| gcmpnl | 1 | LSH | 1.19 | 1.16 | 1.23 | 0.90 |
| mdfnfh | 2 | LSH | 1.21 | 1.18 | 1.25 | 0.86 |
| q60mfh | 3 | LSH | 1.22 | 1.17 | 1.27 | 0.18 |
| q60pfh | 4 | LSH | 1.18 | 1.13 | 1.22 | 0.93 |
| rcimnl | 5 | LSH | 1.26 | 1.22 | 1.30 | 0.97 |
| smannl | 6 | LSH | 1.20 | 1.16 | 1.23 | 0.96 |
| gcmpnl | 1 | LSL | 0.88 | 0.86 | 0.90 | 0.36 |
| mdfnfh | 2 | LSL | 0.86 | 0.84 | 0.88 | 0.81 |
| q60mfh | 3 | LSL | 0.84 | 0.81 | 0.87 | 0.27 |
| q60pfh | 4 | LSL | 0.88 | 0.86 | 0.91 | 0.57 |
| rcimnl | 5 | LSL | 0.86 | 0.84 | 0.88 | 0.41 |
| smannl | 6 | LSL | 0.89 | 0.87 | 0.91 | 0.19 |
| gcmpnl | 1 | CMF | 0.88 | 0.86 | 0.89 | 0.98 |
| mdfnfh | 2 | CMF | 0.88 | 0.86 | 0.90 | 0.99 |
| q60mfh | 3 | CMF | 0.90 | 0.88 | 0.91 | 0.99 |
| q60pfh | 4 | CMF | 0.89 | 0.87 | 0.91 | 0.94 |
| rcimnl | 5 | CMF | 0.89 | 0.87 | 0.91 | 0.95 |
| smannl | 6 | CMF | 0.90 | 0.88 | 0.92 | 0.98 |
| gcmpnl | 1 | CPH | 1.25 | 1.22 | 1.32 | 0.95 |
| mdfnfh | 2 | CPH | 1.21 | 1.18 | 1.25 | 0.42 |
| q60mfh | 3 | CPH | 1.24 | 1.18 | 1.41 | 0.10 |
| q60pfh | 4 | CPH | 1.17 | 1.15 | 1.20 | 0.69 |
| rcimnl | 5 | CPH | 1.26 | 1.22 | 1.36 | 0.80 |
| smannl | 6 | CPH | 1.34 | 1.27 | 1.61 | 0.87 |
| gcmpnl | 1 | CPL | 0.84 | 0.81 | 0.86 | 0.69 |
| mdfnfh | 2 | CPL | 0.82 | 0.80 | 0.85 | 1.00 |
| q60mfh | 3 | CPL | 0.84 | 0.81 | 0.86 | 0.55 |
| q60pfh | 4 | CPL | 0.86 | 0.83 | 0.89 | 0.52 |
| rcimnl | 5 | CPL | 0.85 | 0.83 | 0.87 | 0.91 |
| smannl | 6 | CPL | 0.78 | 0.49 | 0.85 | 0.05 |
| gcmpnl | 1 | HOH | | | | |
| mdfnfh | 2 | HOH | 5.61 | 4.69 | 6.65 | 0.99 |
| q60mfh | 3 | HOH | 4.76 | 3.44 | 5.96 | 0.65 |
| q60pfh | 4 | HOH | 5.53 | 4.45 | 6.73 | 0.38 |
| rcimnl | 5 | HOH | 5.33 | 4.16 | 6.48 | 0.79 |
| smannl | 6 | HOH | 4.92 | 3.97 | 5.95 | 0.82 |
| gcmpnl | 1 | HOL | | | | |
| mdfnfh | 2 | HOL | -4.14 | -5.55 | -1.75 | 0.96 |
| q60mfh | 3 | HOL | -6.87 | -8.22 | -5.29 | 0.55 |
| q60pfh | 4 | HOL | -6.21 | -7.46 | -4.85 | 0.99 |
| rcimnl | 5 | HOL | -5.79 | -7.00 | -4.41 | 0.98 |

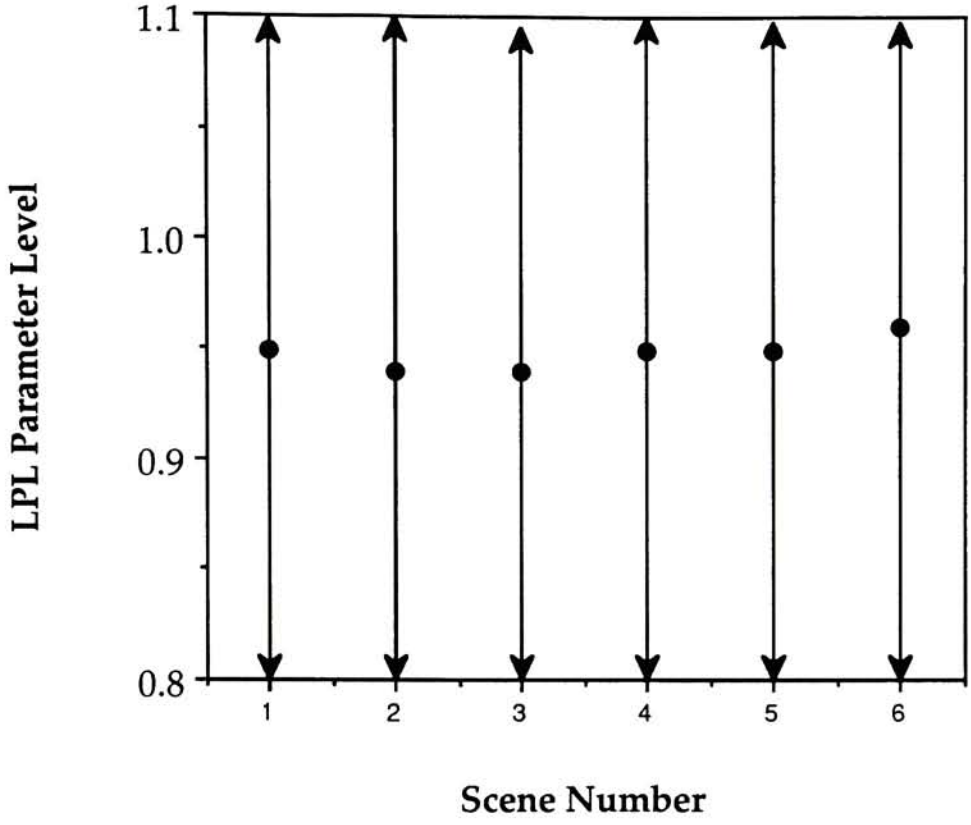
| | | | | | | |
|---|---|---|---|---|---|---|
| smannl | 6 | HOL | -5.32 | -6.48 | -3.96 | 0.92 |



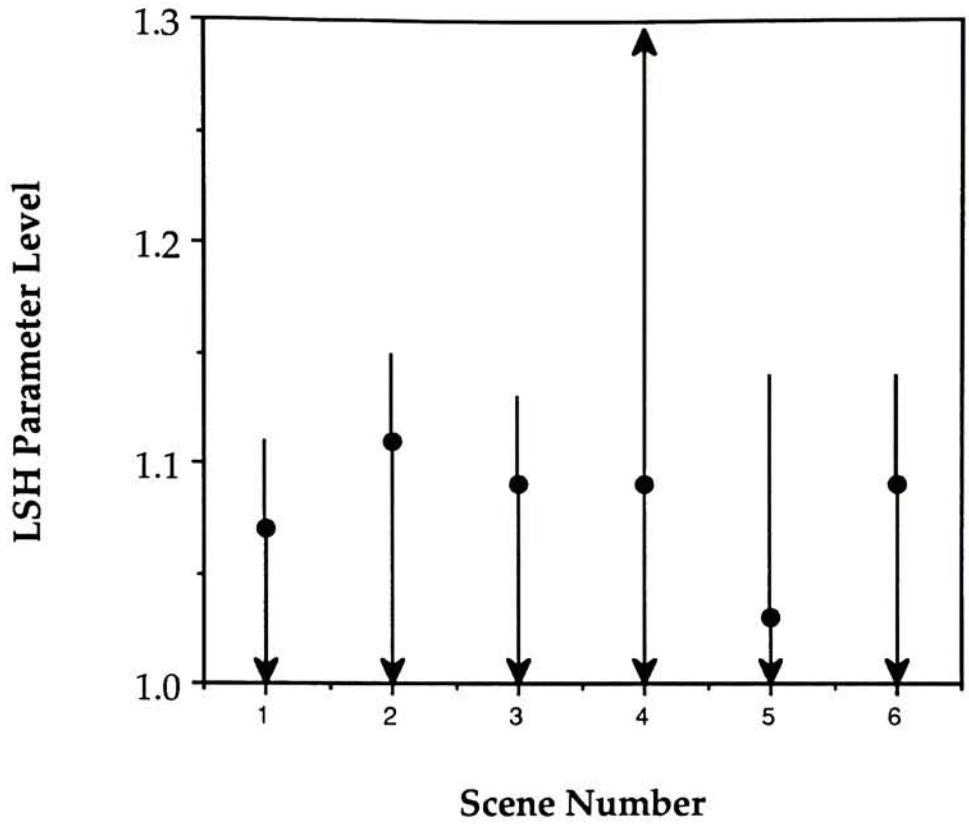

Graph J.1 : Perceptibility by Scene for Lightness Multiplicative Factor



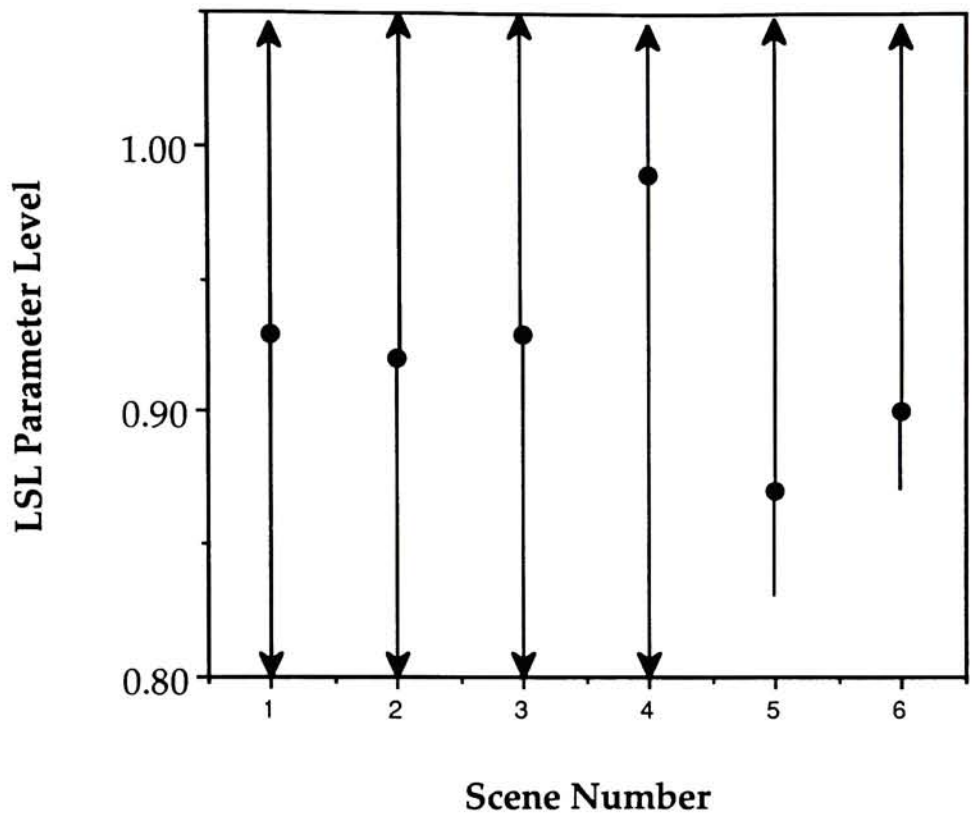

Graph J.2 : Perceptibility by Scene for Lightness High Power Function



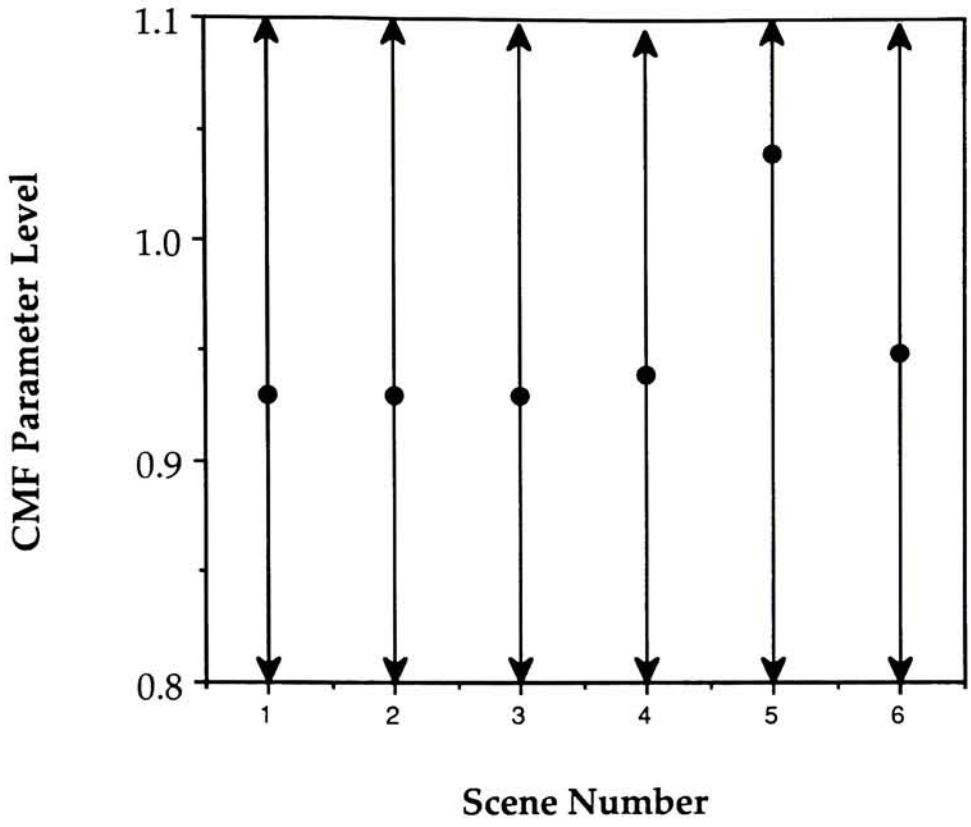

Graph J.3 : Perceptibility by Scene for Lightness Low Power Function



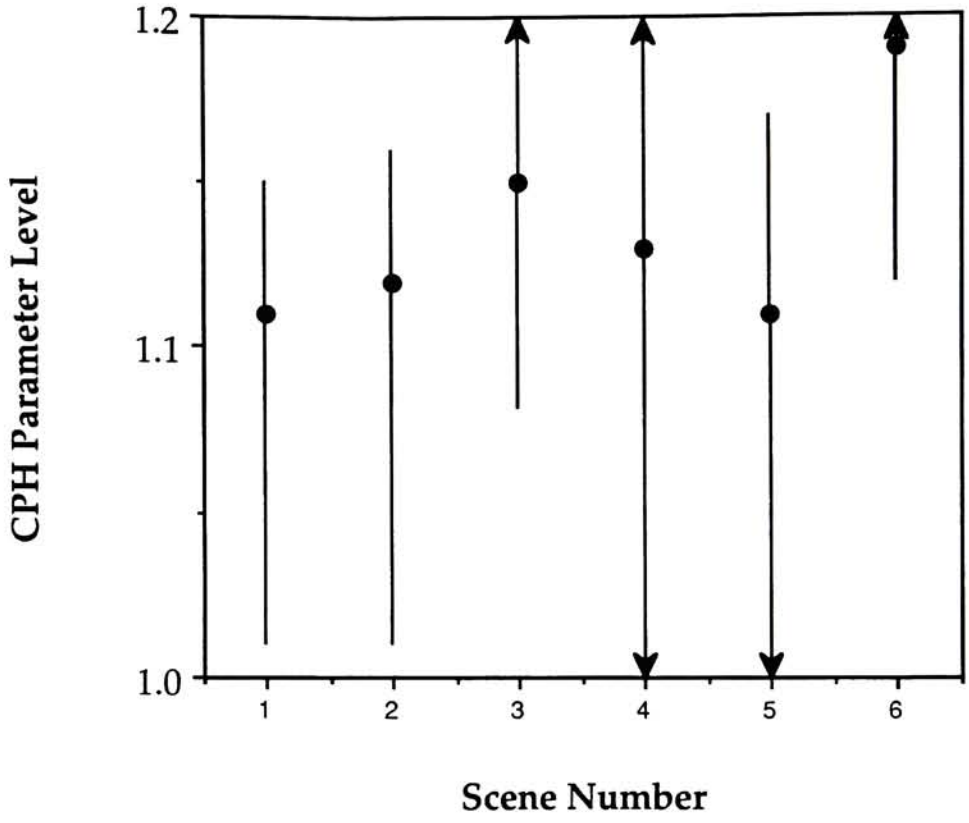

Graph J.4 : Perceptibility by Scene for Lightness High Sigmoidal Function



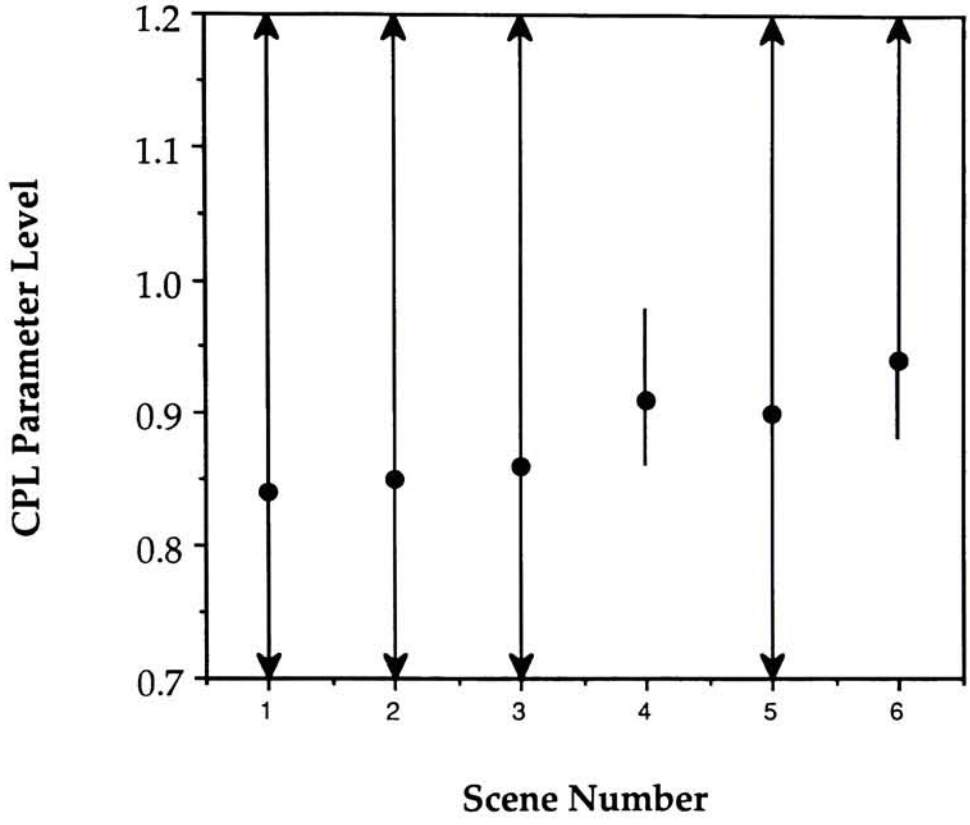

Graph J.5 : Perceptibility by Scene for Lightness Low Sigmoidal Function



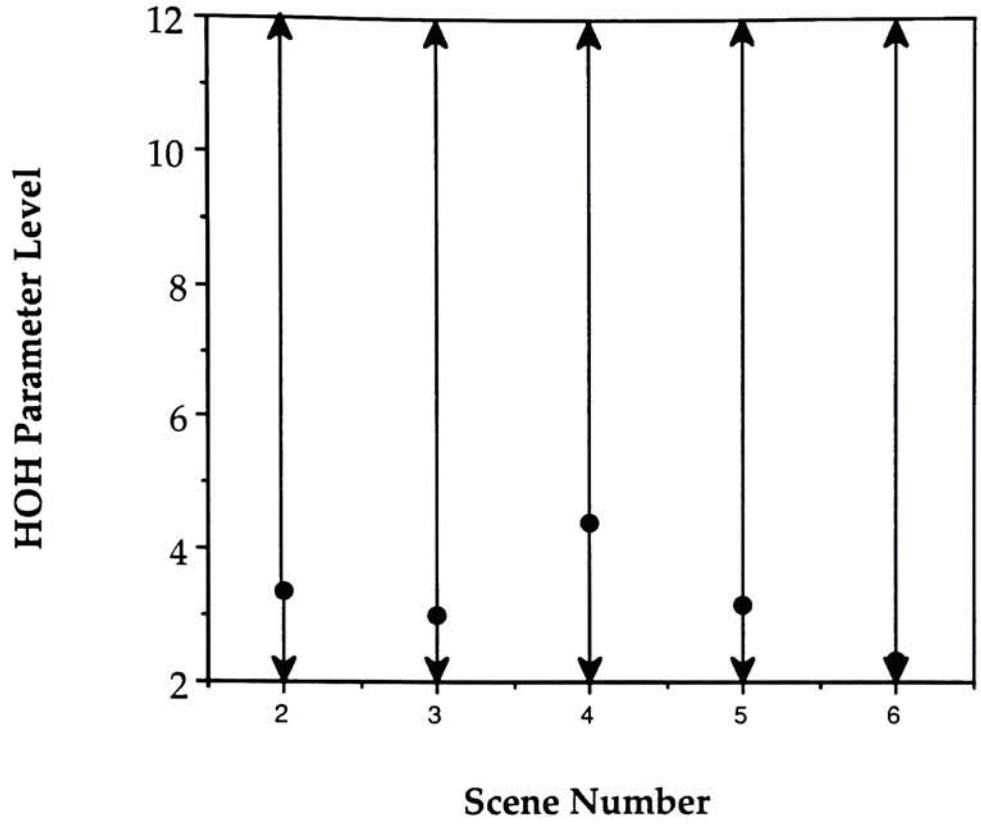

Graph J.6 : Perceptibility by Scene for Chroma Multiplicative Factor



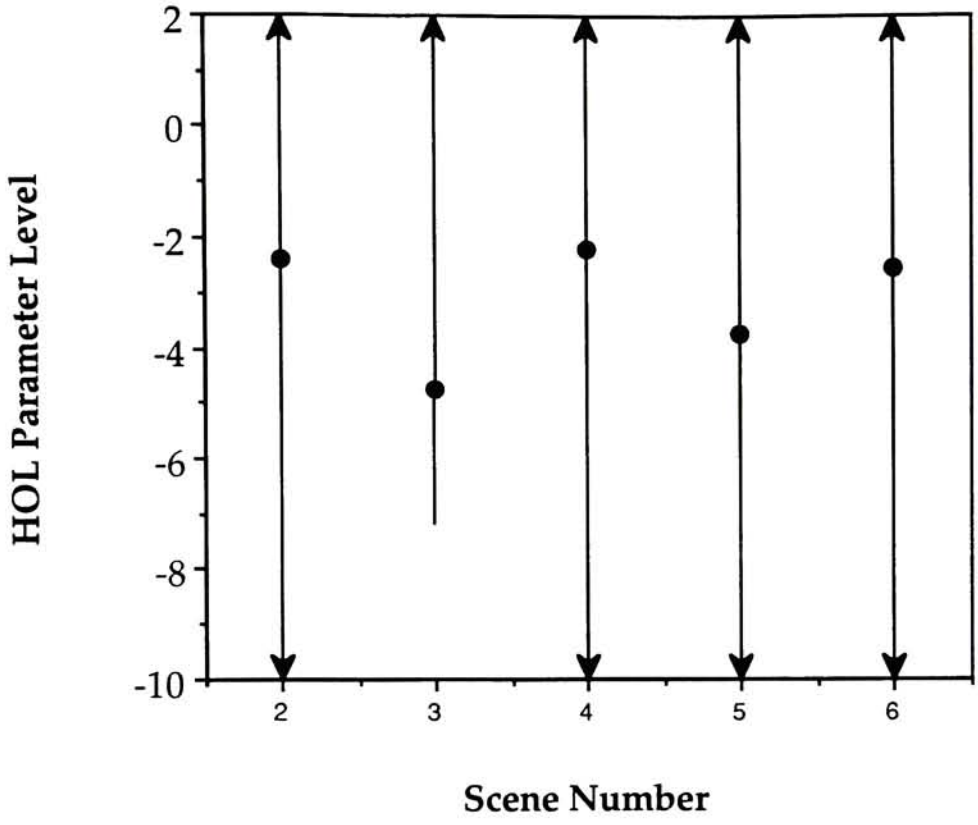

Graph J.7 : Perceptibility by Scene for Chroma High Power Function



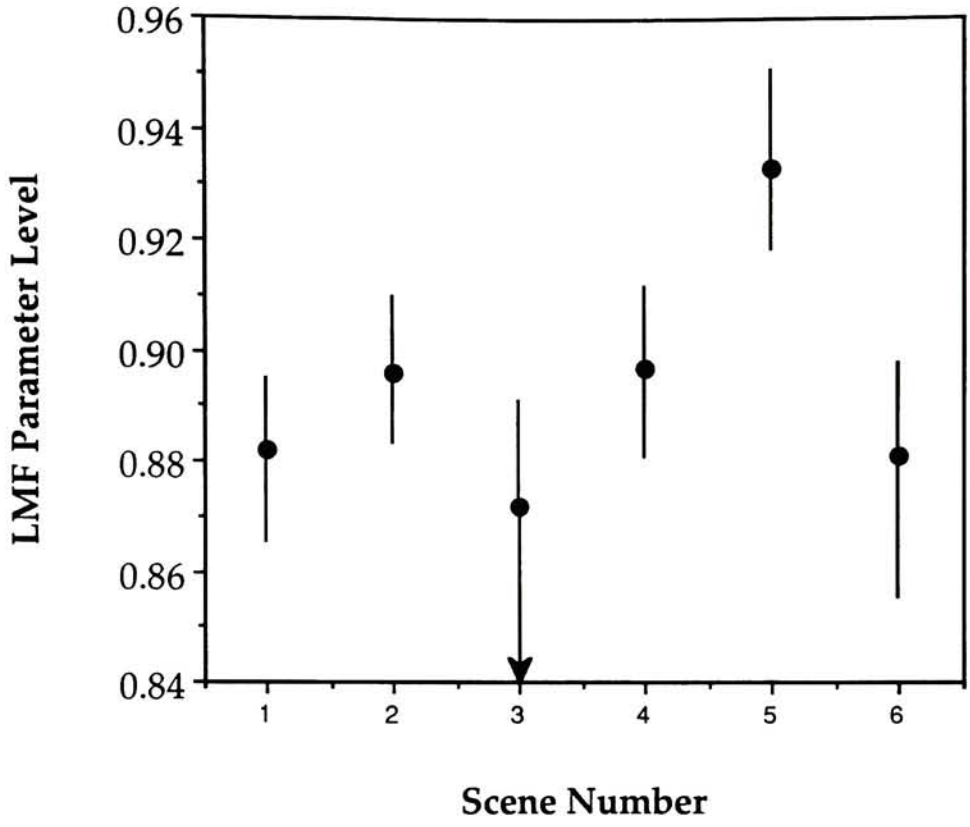

Graph J.8 : Perceptibility by Scene for Chroma Low Power Function



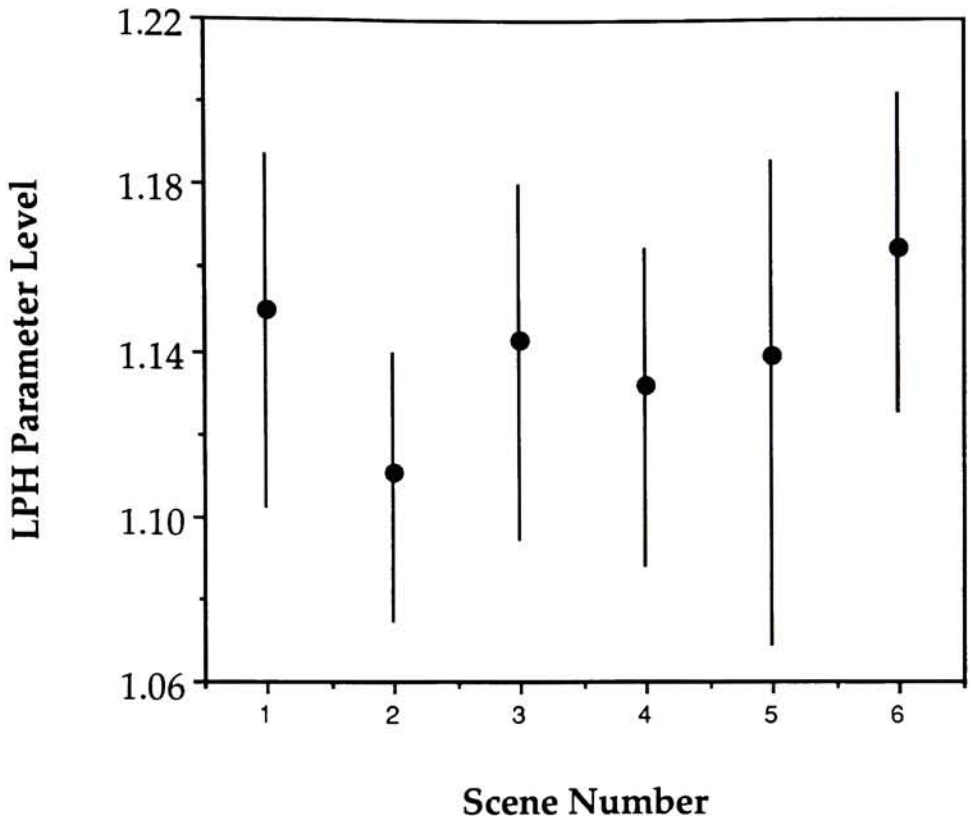

Graph J.9 : Perceptibility by Scene for Hue Angle Positive Additive Offset



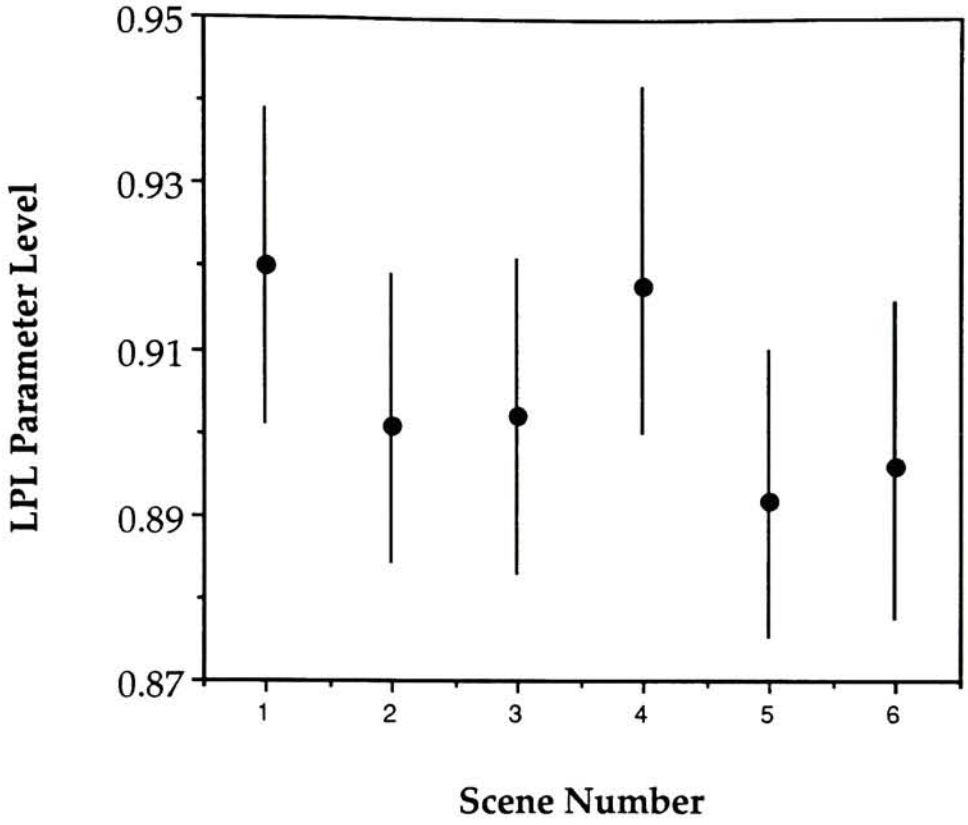

Graph J.10 : Perceptibility by Scene for Hue Angle Negative Additive Offset



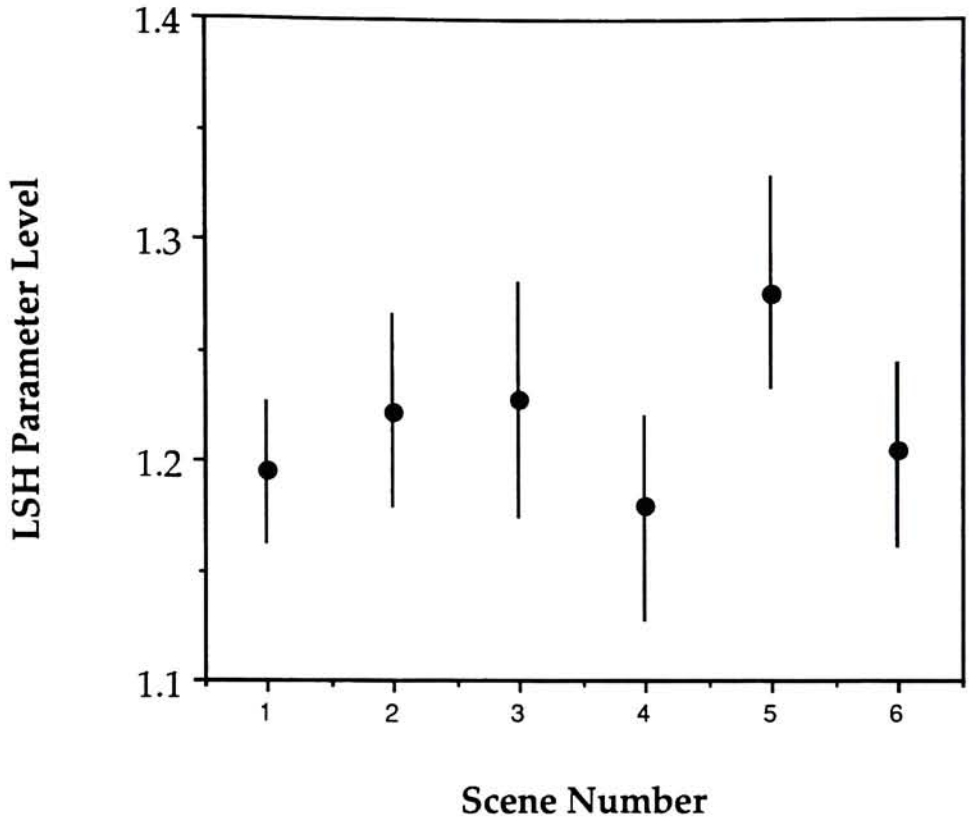

Graph J.11 : Acceptability by Scene for Lightness Multiplicative Factor



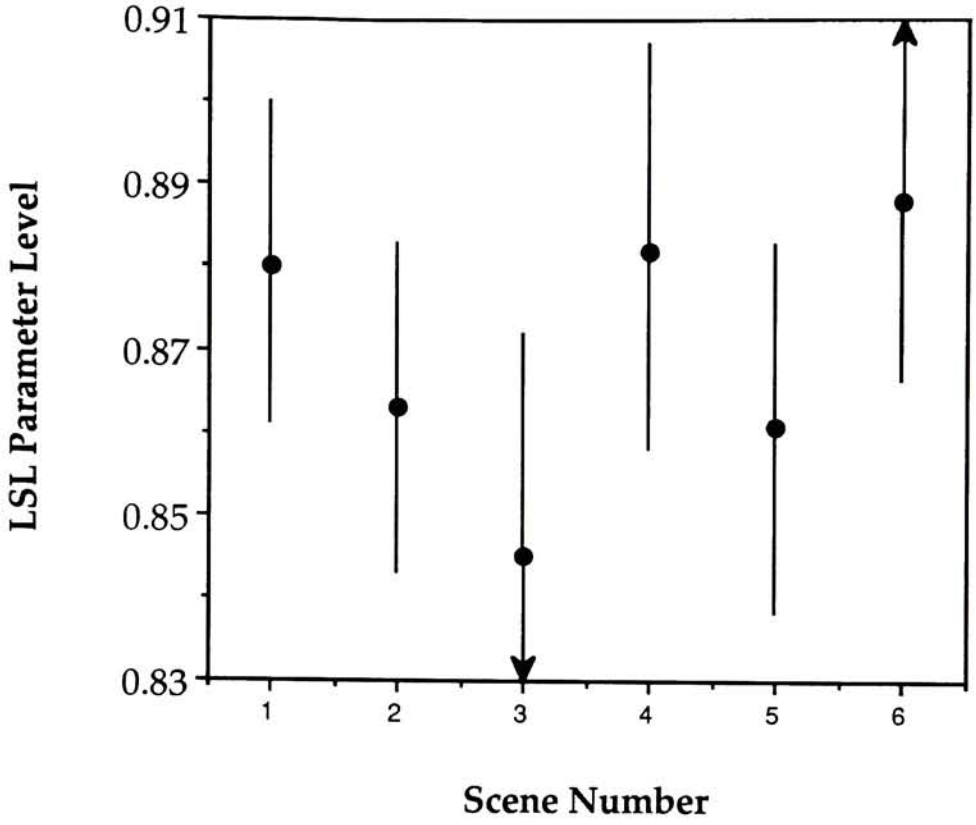

Graph J.12 : Acceptability by Scene for Lightness High Power Function



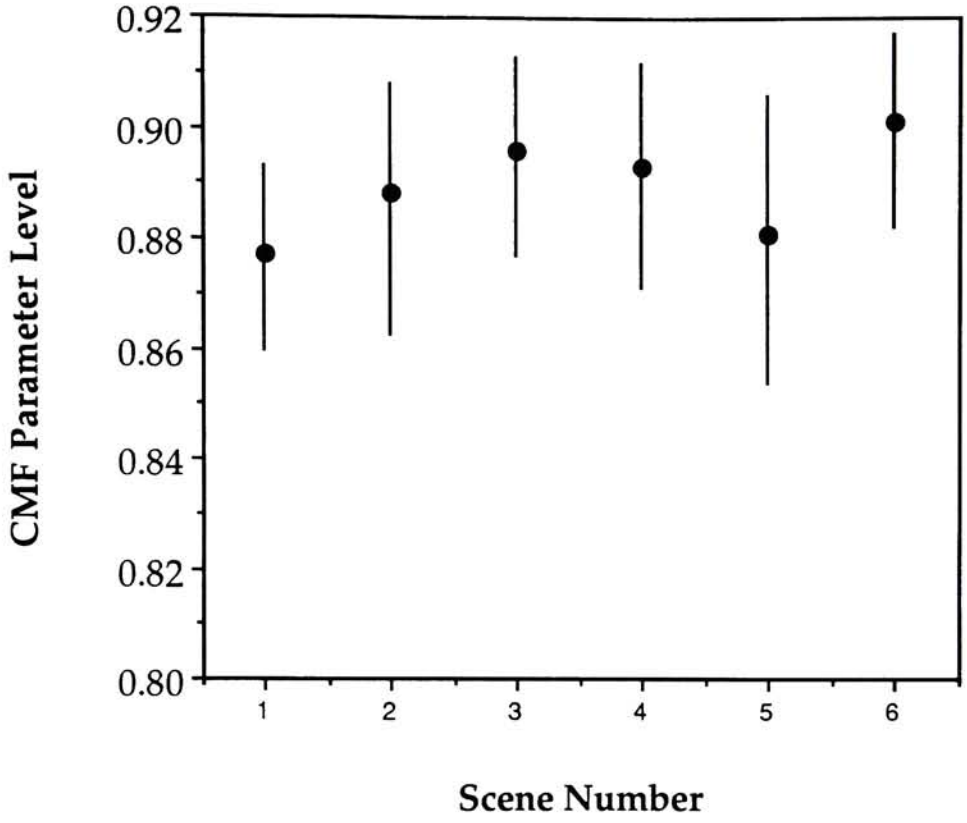

Graph J.13 : Acceptability by Scene for Lightness Low Power Function



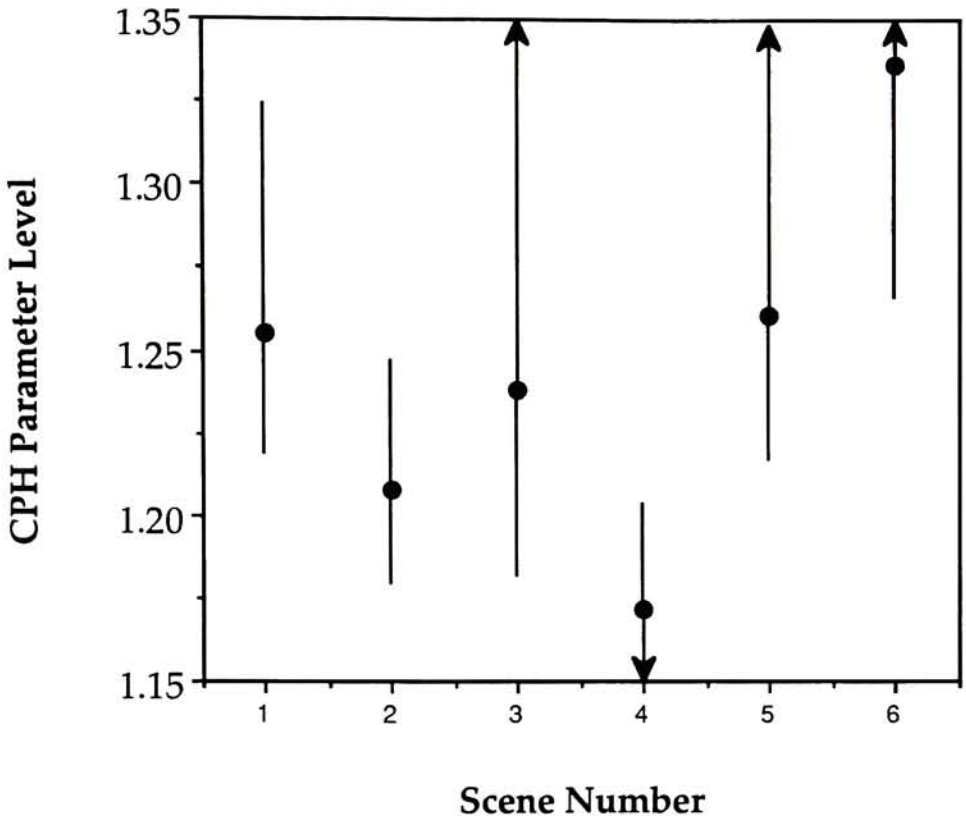

Graph J.14 : Acceptability by Scene for Lightness High Sigmoidal Function



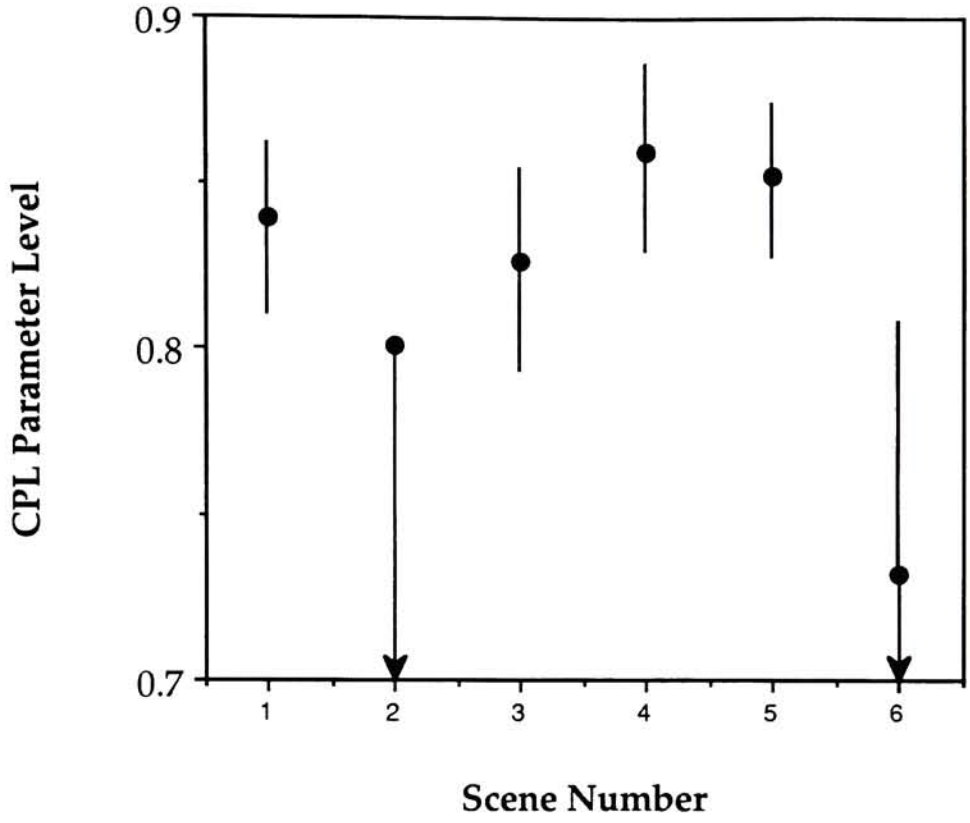

Graph J.15 : Acceptability by Scene for Lightness Low Sigmoidal Function



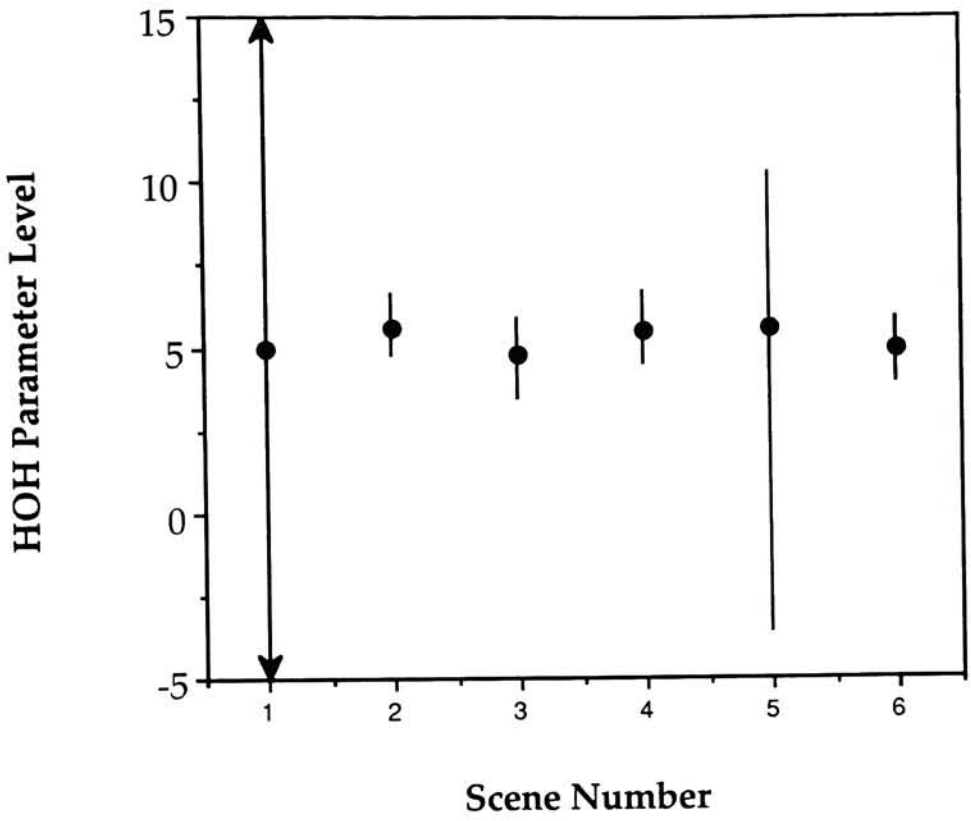

Graph J.16 : Acceptability by Scene for Chroma Multiplicative Factor



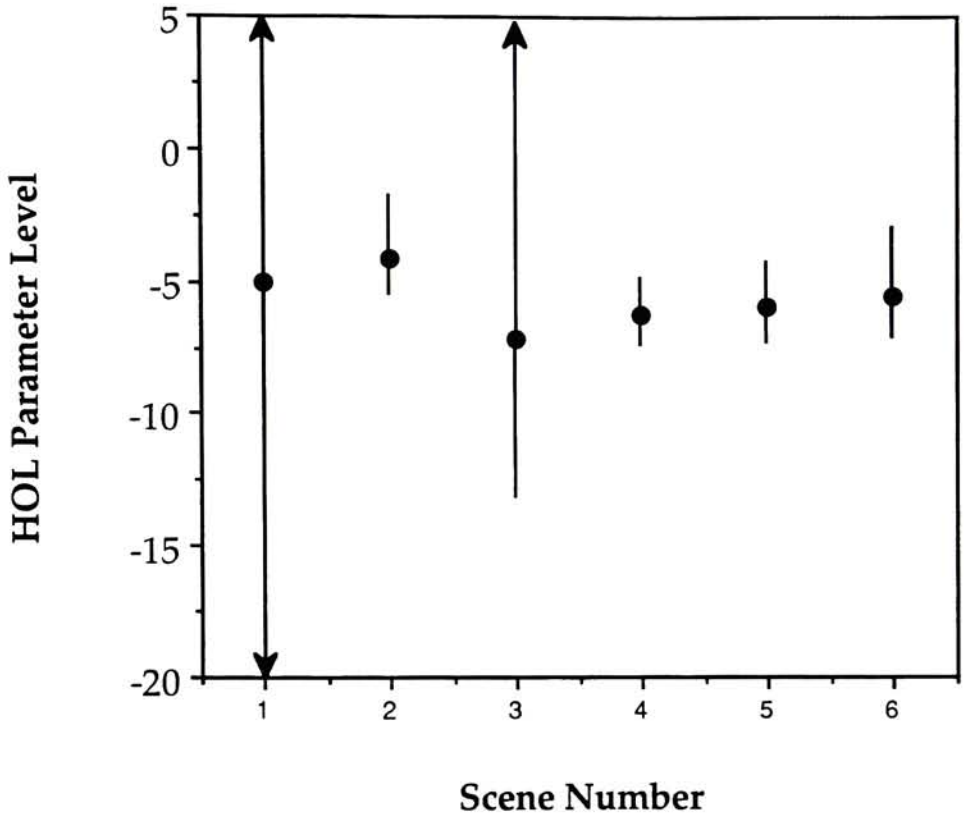

Graph J.17 : Acceptability by Scene for Chroma High Power Function




Graph J.18 : Acceptability by Scene for Chroma Low Power Function



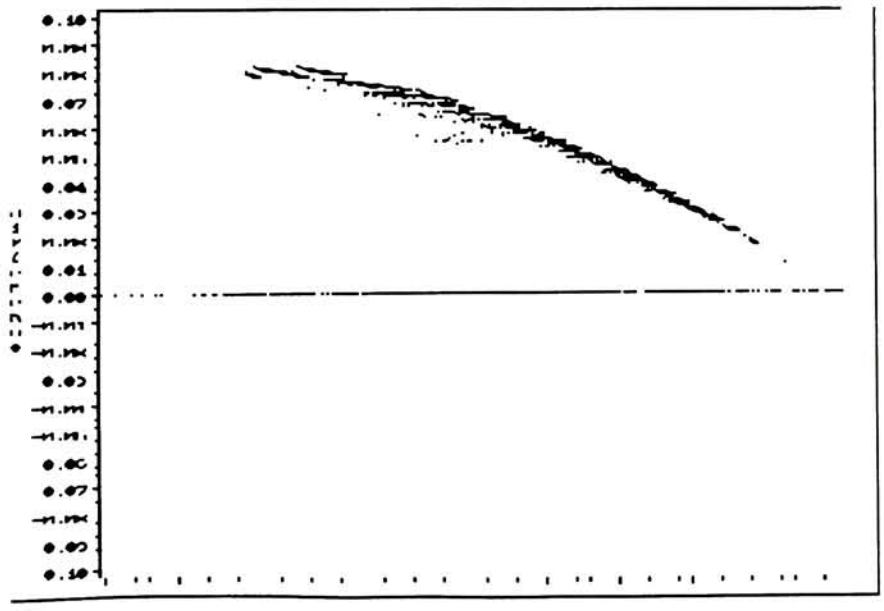

Graph J.19 : Acceptability by Scene for Hue Angle Positive Additive Offset



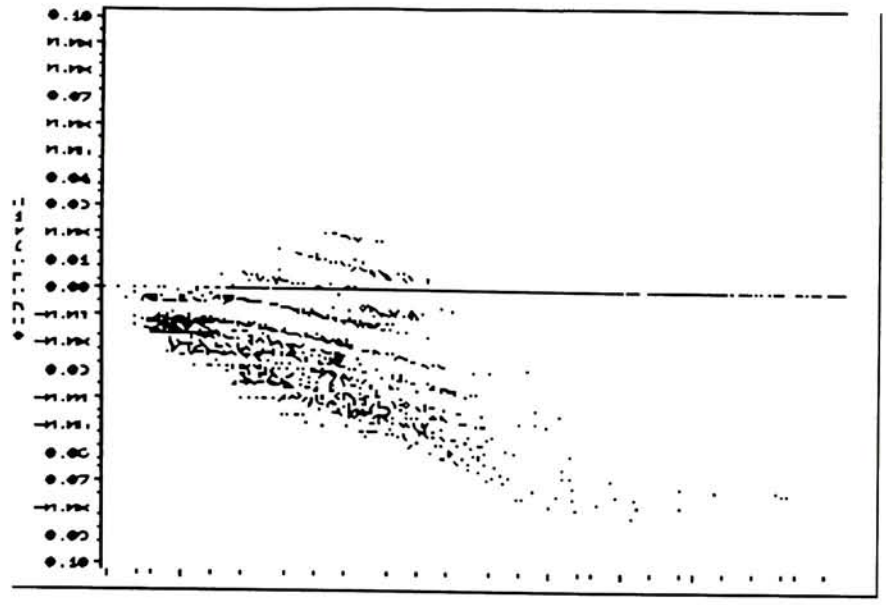

Graph J.20 : Acceptability by Scene for Hue Angle Negative Additive Offset

# Appendix K : Non-Linear Regression Example

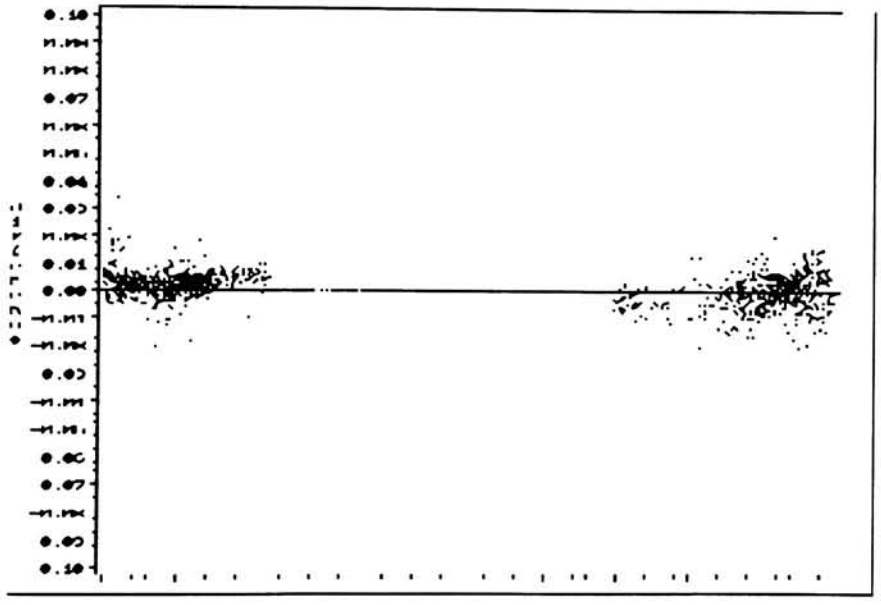
An example of using the tolerances derived in the research is shown below. An original image was manipulated in RGB device space by applying a gamma correction of 1.2 using a simple power function. These images were converted into CIELABL*C*h° space and then into text format for input into the SAS program listed below. The proper transfer function is determined by choosing the highest 'T' values for each dimension. A plot of the resulting data was made to visually determine which dimensions needed to be analyzed. Graphs K.1-K.3 show the difference between the original and manipulated image value for L*, C*, and hue angle on a scale from -0.1 to 0.1 plotted against the original values for each dimension. Hue angle does not contribute to the analysis, although some wrap-around errors between 0 and 360 degrees create some artificial noise.



Graph K.1 : L* error vs. original L* for rgb gamma of 1.2



Graph K.2 : C* error vs. original C* for rgb gamma of 1.2



Graph K.3 : h° error vs. original h° for rgb gamma of 1.2

The lightness and chroma dimensions were regressed using the three transfer functions discussed previously. The statistical significance is determined by evaluating the respective 'T' values. We have initially found

that a 'T' value of 3000 or less indicates a poor model fit. If the 'T' values are greater than 3000, then the transfer function with the highest highest 'T' value is chosen to be analyzed. The parameter estimate from the chosen transfer function regression output is compared against the tolerances in tables 5.2-1 and 5.3-1 for perceptibility and acceptability differences.

In these examples, the lightness power function and the chroma power function best fit the data. The low 'T' values for C* indicate a poor model fit with all of the transfer functions, which is apparent from the noise level in graph K.2. What is significant is that chroma was effectively reduced by a "neutral" manipulation in RGB device space. The L* power function had an acceptable 'T' value with the resulting parameter estimate of 1.22. Comparing this value with tables 5.2-1 and 5.3-1 indicate that the change is both perceptible and unacceptable. It should be noted however, the acceptability estimate is just beyond the fiducial limits. In conclusion, this method provides the ability to judge whether images are perceptibly and acceptably different from the original image with having to view the actual images in an observational environment.

**SAS Code Example :**

```
options device=vga nocenter nodate pagesize=60 linesize=78;libname hdd
"d:\mds\sas\thesis\nlin";

data gamma;
  infile "gamma.txt";
 input dl dc dh;
 dl = dl / 100.0;
 dc = dc / 55.0;
 dh = dh / 360.0;
 run;

data orig;
  infile "risk.txt";
 input ol oc oh;
 ol = ol / 100.0;
 oc = oc / 55.0;
 oh = oh / 360.0;
 run;

data hdd.gam;
 merge orig gamma;
 keep ol dl oc dc oh dh;
 run;

proc model data=hdd.gam;
 title "gamma = 1.2 evaluated with lightness power function";
 var ol dl;
 parms power;
 if (dl = 0.0) then ol = 0.0;
 else ol = exp(power*(log(dl)));
 fit ol start=(power 1);
 run;

proc model data=hdd.gam;
 title "gamma = 1.2 evaluated with lightness sigmoidal function";
 var ol dl;
 parms sigmoid;
 if (dl = 0.0) then ol = 0.0;
 else if (dl = 0.5) then ol = 0.5;
 else if (dl = 1.0) then ol = 1.0;
 else if (dl < 0.5) then
   ol = 0.5 * exp(sigmoid*(log(2*dl)));
```

```
  else
    ol = 0.5 * (1.0 + exp(sigmoid*(log((2*dl)-1.0))));
  fit dl start=(sigmoid 1);
  run;

proc model data=hdd.gam;
  title "gamma =1.2 evaluated with lightness multiplicative factor";
  var ol dl;
  parms mf;
  ol = mf*dl;
  fit ol start=(mf 1);
  run;

proc model data=hdd.gam;
  title "gamma = 1.2 evaluated with chroma power function";
  var oc dc;
  parms power;
  if (dc = 0.0) then oc = 0.0;
  else oc = exp(power*(log(dc)));
  fit oc start=(power 1);
  run;

proc model data=hdd.gam;
  title "gamma = 1.2 evaluated with chroma sigmoidal function";
  var oc dc;
  parms sigmoid;
  if (dc = 0.0) then oc = 0.0;
  else if (dc = 0.5) then oc = 0.5;
  else if (dc = 1.0) then oc = 1.0;
  else if (dc < 0.5) then
    oc = 0.5 * exp(sigmoid*(log(2*dc)));
  else
    oc = 0.5 * (1.0 + exp(sigmoid*(log((2*dc)-1.0))));
  fit dc start=(sigmoid 1);
  run;

proc model data=hdd.gam;
  title "gamma =1.2 evaluated with chroma multiplicative factor";
  var oc dc;
  parms mf;
  oc = mf*dc;
  fit oc start=(mf 1);
  run;
```

**Abbreviated SAS Listing from the above example**

**RGB gamma = 1.2 evaluated with lightness power function** **1**
Nonlinear OLS Summary of Residual Errors

| Equation | DF Model | DF Error | SSE | MSE | Root MSE | R-Square | Adj R-Sq |
|---|---|---|---|---|---|---|---|
| OL | 1 | 11007 | 0.19993 | 0.00001816 | 0.0042619 | 0.9995 | 0.9995 |

Nonlinear OLS Parameter Estimates

| Parameter | Estimate | Approx. Std Err | 'T' Ratio | Approx. Prob>\|T\| |
|---|---|---|---|---|
| POWER | 1.221045 | 0.0001685 | 7246.83 | 0.0 |

**RGB gama = 1.2 evaluated with lightness sigmoidal function** **5**
Nonlinear OLS Summary of Residual Errors

| Equation | DF Model | DF Error | SSE | MSE | Root MSE | R-Square | Adj R-Sq |
|---|---|---|---|---|---|---|---|
| OL | 1 | 11007 | 9.89824 | 0.0008993 | 0.02999 | 0.9753 | 0.9753 |

Nonlinear OLS Parameter Estimates

| Parameter | Estimate | Approx. Std Err | 'T' Ratio | Approx. Prob>\|T\| |
|---|---|---|---|---|
| SIGMOID | 1.394558 | 0.0026816 | 520.04 | 0.0 |

**RGB gamma = 1.2 evaluated with lightness multiplicative factor** **12**
Nonlinear OLS Summary of Residual Errors

| Equation | DF Model | DF Error | SSE | MSE | Root MSE | R-Square | Adj R-Sq |
|---|---|---|---|---|---|---|---|
| OL | 1 | 11007 | 10.96758 | 0.0009964 | 0.03157 | 0.9726 | 0.9726 |

Nonlinear OLS Parameter Estimates

| Parameter | Estimate | Approx. Std Err | 'T' Ratio | Approx. Prob>\|T\| |
|---|---|---|---|---|
| MF | 0.926208 | 0.0004616 | 2006.48 | 0.0 |

**RGB gamma = 1.2 evaluated with chroma power function** 15

Nonlinear OLS Summary of Residual Errors

| Equation | DF Model | DF Error | SSE | MSE | Root MSE | R-Square | Adj R-Sq |
|---|---|---|---|---|---|---|---|
| OC | 1 | 11007 | 3.22698 | 0.0002932 | 0.01712 | 0.9835 | 0.9835 |

Nonlinear OLS Parameter Estimates

| Parameter | Estimate | Approx. Std Err | 'T' Ratio | Approx. Prob>\|T\| |
|---|---|---|---|---|
| POWER | 1.221045 | 0.0005056 | 1861.26 | 0.0 |

**RGB gama = 1.2 evaluated with chroma sigmoidal function** 20

Nonlinear OLS Summary of Residual Errors

| Equation | DF Model | DF Error | SSE | MSE | Root MSE | R-Square | Adj R-Sq |
|---|---|---|---|---|---|---|---|
| OC | 1 | 11007 | 3.55322 | 0.0003228 | 0.01797 | 0.9818 | 0.9818 |

Nonlinear OLS Parameter Estimates

| Parameter | Estimate | Approx. Std Err | 'T' Ratio | Approx. Prob>\|T\| |
|---|---|---|---|---|
| SIGMOID | 0.892348 | 0.0010007 | 891.71 | 0.0 |

**RGB gamma = 1.2 evaluated with chroma multiplicative factor** 24

Nonlinear OLS Summary of Residual Errors

| Equation | DF Model | DF Error | SSE | MSE | Root MSE | R-Square | Adj R-Sq |
|---|---|---|---|---|---|---|---|
| OC | 1 | 11007 | 3.20781 | 0.0002914 | 0.01707 | 0.9836 | 0.9836 |

Nonlinear OLS Parameter Estimates

| Parameter | Estimate | Approx. Std Err | 'T' Ratio | Approx. Prob>\|T\| |
|---|---|---|---|---|
| MF | 1.076206 | 0.0006793 | 1584.25 | 0.0 |